(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平８－３３９３２５

(43)公開日　平成８年（１９９６）１２月２４日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
G06F 12/02	570		G06F 12/02	570	A
9/455		9189-5B	9/44	310	A

審査請求　未請求　請求項の数２８　ＯＬ　（全２８頁）

(21)出願番号　特願平８－１２４３７５

(22)出願日　平成８年（１９９６）５月２０日

(31)優先権主張番号　４８０１０５

(32)優先日　１９９５年６月７日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　３９０００９５３１
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国１０５０４、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　ジョン・ダブル・ゴェッツ
アメリカ合衆国０５４６５、ヴァーモント州ジェリコバターカップ　レイン　３

(74)代理人　弁理士　合田　潔　（外２名）

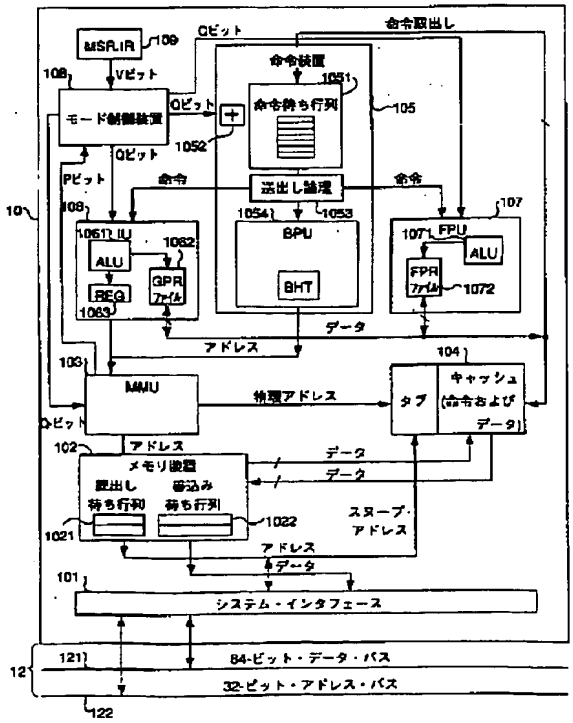
最終頁に続く

(54)【発明の名称】二つの別個の命令セット・アーキテクチャへの拡張をサポートすることができるアーキテクチャ・モード制御を備えたマイクロプロセッサ

(57)【要約】　（修正有）

【課題】　一つの多重タスク処理オペレーティング・システムの下で二つの別個のアーキテクチャをサポートするプロセッサを提供する。

【解決手段】　マイクロプロセッサは、それが作動するところのアーキテクチャコンテキストを制御するモード制御装置を含み、制御装置は、モード・ビットの制御の下でアドレス変換が生じることを許容し、変換機構を一方のアーキテクチャからもう一方のアーキテクチャに切り換えることができるメモリ管理装置（ＭＭＵ）を持っている。一つのＭＭＵが、二つの別個のアーキテクチャのアドレスを変換して、命令が適切にデコードされるようにする。ＭＭＵはまた、一方のアーキテクチャのアドレス変換を他方のそれに対してマッピングして、両方のアーキテクチャ用に書かれたソフトウェアを一つのオペレーティング・システムの制御の下で多重タスク処理することができるようにする。



1

【特許請求の範囲】

【請求項１】別個の命令セットおよびメモリ管理方式を有する第一および第二のアーキテクチャをサポートし、一つの多重タスク処理オペレーティング・システムの下で作動するプロセッサにおいて、

前記第一のアーキテクチャの第一の命令セットの命令をデコードし、前記第二のアーキテクチャの第二の命令セットの命令を直接デコードし、前記第一の命令セットのデコードされた命令を前記第二の命令セットの一つ以上の命令に対してマッピングするための命令セット管理手段と、

前記第一および第二のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行するためのメモリ管理手段と、

メモリから読み出されるプログラムのアーキテクチャ・コンテキストを、前記第一のアーキテクチャのコードまたは前記第二のアーキテクチャのコードのいずれかとして検出し、前記命令セット管理手段および前記メモリ管理手段を制御して、前記第一のアーキテクチャのアドレス変換と第二のアーキテクチャのアドレス変換との間で動的に切り換え、前記第二のアーキテクチャの、一つ以上のマッピングされてデコードされた命令または直接デコードされた命令を実行するための制御手段と、を含むことを特徴とするプロセッサ。

【請求項２】前記メモリ管理手段が、実効アドレスを仮想アドレスに変換する際に第一または第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するための、前記制御手段によって制御されるモード制御機構を含む請求項１記載のプロセッサ。

【請求項３】前記モード制御機構が前記プロセッサの命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が、前記命令セット管理手段による適切なデコードのために取り出され、配列されるようにする請求項２記載のプロセッサ。

【請求項４】前記メモリ管理装置が、前記第二のアーキテクチャのページ・テーブル項目からページ・モード制御ビットを読み出し、前記ページ・モード制御ビットが前記制御手段に供給されて、前記メモリ管理手段によるアドレス変換を制御する請求項３記載のプロセッサ。

【請求項５】前記制御手段が、

プロセッサがどちらのアーキテクチャ・コンテキストの下で作動するかを制御するためのアーキテクチャ・コンテキスト制御機構であって、実効アドレスを仮想アドレスに変換する際に前記プロセッサの前記メモリ管理装置によって使用されるアーキテクチャ変換方法を制御し、さらに、前記プロセッサの命令取出しおよびデコード論理を制御し、さらに、プロセスおよび構築されたリソースの実行コンテキストを制御し、さらに、前記プロセッサの割込みおよび例外機構を制御するためのアーキテクチャ・コンテキスト制御機構と、

2

一つのアドレス変換機構が、一方のアーキテクチャのアドレスをもう一方のアーキテクチャのアドレスの変換に対してマッピングすることを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびの保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするための修飾モード制御機構と、を含む請求項１記載のプロセッサ。

【請求項６】前記命令セット管理手段が、

前記第一の命令セットの命令をデコードするための第一のデコード手段と、

前記第二の命令セットの命令をデコードするための第二のデコード手段と、

前記第一のデコード手段からのデコードされた命令を前記第二の命令セットの一つ以上のデコードされた命令に対してマッピングするためのマッピング手段と、

前記マッピング手段からのデコードされた命令または前記第二のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される選択手段と、を含む請求項１記載のプロセッサ。

【請求項７】前記第一および第二のデコード手段が前記第一および第二のアーキテクチャの簡単な命令をデコードし、簡単な命令が、基本的な演算クラスの中にマッピングし、一つの実行装置によって取り扱われることができる命令であり、前記命令管理手段がさらに、

前記第一の命令セットの複雑な命令をデコードするための第三のデコード手段と、

前記第二の命令セットの複雑な命令をデコードするための第四のデコード手段と、

前記第三または第四のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される第二の選択手段と、

前記第三または第四のデコード手段からのデコードされた命令を前記第二の命令セットの多数の簡単な命令に対してマッピングするための、前記第二の選択手段の出力を受ける第二のマッピング手段と、

複雑な命令がデコードされる場合には前記第二のマッピング手段の出力を選択し、簡単な命令がデコードされる場合には前記第一の選択手段の出力を選択するための、前記第三または第四のデコード手段の一方からの有効な信号に応答する第三の選択手段と、をさらに含む請求項６記載のプロセッサ。

【請求項８】前記制御装置が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前

記プロセッサが複数の状態の一つに入るように制御するモード制御装置を含む請求項１記載のプロセッサ。

【請求項９】前記プロセッサが、主メモリから読み出される命令のページ・モードを検出し、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを前記第二のビットによって前記モード制御装置に知らせるメモリ管理装置を含む請求項８記載のプロセッサ。

【請求項１０】前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする請求項８記載のプロセッサ。

【請求項１１】一つの多重タスク処理オペレーティング・システムの下で作動するプロセッサにおいて具現化される、二つの別個の命令セット・アーキテクチャをサポートする方法であって、

ソフトウェアによって供給されるモード制御ビットに応答して実効アドレスを仮想アドレスに変換する際に第一および第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するステップと、

プロセッサの命令取出しおよびデコード機構を制御して、二つの異なるアーキテクチャの命令がソフトウェアによって供給されるモード制御ビットに応答して適切にデコードされるようにするステップと、

一方のアーキテクチャの変換をもう一方のアーキテクチャのそれに対してマッピングすることにより、前記二つの異なるアーキテクチャのアドレスを変換するステップと、

多重タスク処理環境において、一方のアーキテクチャ用に書かれたアプリケーション・ソフトウェアからもう一方のアーキテクチャ用に書かれたアプリケーション・ソフトウェアに切り換えるステップと、を含むことを特徴とする方法。

【請求項１２】マイクロプロセッサの割込みおよび例外機構を制御するステップと、

一つのアドレス変換機構が、第一のアーキテクチャのアドレスを第二のアーキテクチャの変換に対してマッピングすることによって第一のアーキテクチャのアドレスを変換することを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびに保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするステップと、

命令が実行すべきところのアーキテクチャ・コンテキストを決定するステップと、

をさらに含む請求項１１記載のプロッセッサにおいて具現化される方法。

【請求項１３】二つの別個の命令セット・アーキテクチャをサポートし、一つの多重タスク処理オペレーティング・システムの下で作動するマイクロプロセッサと、

前記二つのアーキテクチャ用のアプリケーション・ソフトウェアを記憶する外部メモリ・デバイスと、

前記マイクロプロセッサを前記外部メモリ・デバイスに接続するシステム・バスとを含み、

前記マイクロプロセッサが、前記システム・バスに接続された内部バスを有し、前記マイクロプロセッサが、

前記第一のアーキテクチャの第一の命令セットの命令をデコードし、前記第二のアーキテクチャの第二の命令セットの命令を直接デコードし、第一の命令セットのデコードされた命令を前記第二の命令セットの一つ以上の命令に対してマッピングするための命令セット管理手段と、

前記第一および第二のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行するためのメモリ管理手段と、

メモリから読み出されるプログラムのアーキテクチャ・コンテキストを、前記第一のアーキテクチャのコードまたは前記第二のアーキテクチャのコードのいずれかとして検出し、前記命令セット管理手段および前記メモリ管理手段を制御して、前記第一のアーキテクチャのアドレス変換と前記第二のアーキテクチャのアドレス変換との間で動的に切り換え、前記第二のアーキテクチャの、一つ以上のマッピングされてデコードされた命令または直接デコードされた命令を実行するための制御手段と、を含むことを特徴とするコンピュータ・システム。

【請求項１４】前記マイクロプロセッサの前記メモリ管理手段が、実効アドレスを仮想アドレスに変換する際に第一または第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するためのモード制御機構を含む請求項１３記載のコンピュータ・システム。

【請求項１５】前記モード制御機構が前記プロセッサの命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が、前記命令セット管理手段による適切なデコードのために取り出され、配列されるようにする請求項１４記載のコンピュータ・システム。

【請求項１６】前記メモリ管理装置が、前記第二のアーキテクチャのページ・テーブル項目からページ・モード制御ビットを読み出し、前記ページ・モード制御ビットが前記制御手段に供給されて、前記メモリ管理手段によるアドレス変換を制御する請求項１５記載のコンピュータ・システム。

【請求項１７】前記制御手段が、

プロセッサがどちらのアーキテクチャ・コンテキストの

下で作動するかを制御するためのアーキテクチャ・コンテキスト制御機構であって、実効アドレスを仮想アドレスに変換する際に前記マイクロプロセッサのメモリ管理装置によって使用されるアーキテクチャ変換方法を制御し、さらに、前記マイクロプロセッサの命令取出しおよびデコード論理を制御し、さらに、プロセスおよび構築されたリソースの実行コンテキストを制御し、さらに、前記マイクロプロセッサの割込みおよび例外機構を制御するためのアーキテクチャ・コンテキスト制御機構と、一つのアドレス変換機構が、一方のアーキテクチャのアドレスをもう一方のアーキテクチャのアドレス変換に対してマッピングすることを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびに保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするための修飾モード制御機構と、を含む請求項１３記載のコンピュータ・システム。

【請求項１８】前記マイクロプロセッサの前記命令セット管理手段が、

前記第一の命令セットの命令をデコードするための第一のデコード手段と、

前記第二の命令セットの命令をデコードするための第二のデコード手段と、

前記第一のデコード手段からのデコードされた命令を前記第二の命令セットの一つ以上のデコードされた命令に対してマッピングするためのマッピング手段と、

前記マッピング手段からのデコードされた命令または前記第二のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される選択手段と、を含む請求項１３記載のコンピュータ・システム。

【請求項１９】前記第一および第二のデコード手段が前記第一および第二のアーキテクチャの簡単な命令をデコードし、簡単な命令が、基本的な演算クラスの中にマッピングし、一つの実行装置によって取り扱われることができる命令であり、前記命令管理手段がさらに、

前記第一の命令セットの複雑な命令をデコードするための第三のデコード手段と、

前記第二の命令セットの複雑な命令をデコードするための第四のデコード手段と、

前記第三または第四のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される第二の選択手段と、

前記第三または第四のデコード手段からのデコードされた命令を前記第二の命令セットの多数の簡単な命令に対してマッピングするための、前記第二の選択手段の出力を受ける第二のマッピング手段と、

複雑な命令がデコードされる場合には前記第二のマッピング手段の出力を選択し、簡単な命令がデコードされる場合には前記第一の選択手段の出力を選択するための、前記第三または第四のデコード手段の一方からの有効な信号に応答する第三の選択手段と、を含む請求項１８記載のコンピュータ・システム。

【請求項２０】前記マイクロプロセッサの前記制御手段が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前記プロセッサが複数の状態の一つに入るように制御するモード制御装置を含む請求項１３記載のコンピュータ・システム。

【請求項２１】前記プロセッサが、主メモリから読み出される命令のページ・モードを検出し、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを前記第二のビットによって前記モード制御装置に知らせるメモリ管理装置を含む請求項２０記載のコンピュータ・システム。

【請求項２２】前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする請求項２０記載のコンピュータ・システム。

【請求項２３】前記マイクロプロセッサが、サポートされる二つのアーキテクチャに共通のフォーマットを使用して具現化される一つのメモリ管理装置を有している請求項１３記載のコンピュータ・システム。

【請求項２４】前記マイクロプロセッサが、前記サポートされる二つのアーキテクチャによって共用される一つの命令取出し機構と、別々の命令デコード機構とを有している請求項１３記載のコンピュータ・システム。

【請求項２５】前記マイクロプロセッサのすべての実行リソースが前記サポートされる二つのアーキテクチャに共通である請求項１３記載のコンピュータ・システム。

【請求項２６】別個の命令セットおよびメモリ管理方式を有する第一および第二のアーキテクチャをサポートするプロセッサにおいて、

前記第一および第二のアーキテクチャ用に書かれたアプリケーション・プログラムを記憶する外部メモリに接続するためのシステム・インタフェースと、

前記第二のアーキテクチャのページ・テーブル項目からページ・モード制御ビットを読み出すための、前記システム・インタフェースに接続されたメモリ管理装置と、

第一の命令セットの命令をデコードし、デコードされた命令を第二の命令セットの一つ以上のデコードされた命

令に対してマッピングするための第一のデコード手段と、第二の命令セットの命令をデコードするための第二のデコード手段と、前記第一または第二のデコード手段によってデコードされた命令を選択するための選択手段とを含む取出しおよびデコード機構を含む命令装置と、第一および第二のアーキテクチャ・モードのどちらを可能にするかに依存してアクセスすることができる複数のレジスタを有する、前記命令装置からのデコードされた命令を受信し、実行するために接続された実行装置と、前記命令装置および前記実行装置に接続されたモード制御装置とを含み、前記メモリ管理装置によって読み出される前記ページ・モード制御ビットが前記モード制御装置に供給され、前記第一または第二のアーキテクチャ・モードのいずれかを可能にしてアドレス変換を制御し、前記プロセッサの前記命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が適切なデコードに備えて取り出され、配列されるようにすることを特徴とするプロセッサ。

【請求項２７】前記モード制御装置が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前記プロセッサが複数の状態の一つに入るように制御する請求項２６記載のプロセッサ。

【請求項２８】前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする請求項２７記載のプロセッサ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は一般にマイクロプロセッサに関し、詳細には、二つの別個の命令セット・アーキテクチャ（以下、単に「アーキテクチャ」という）に対する拡張をサポートすることができるアーキテクチャ・モード制御を備えたマイクロプロセッサに関する。

【０００２】

【従来の技術】現在、パーソナル・コンピュータには、競合する二つのマイクロプロセッサ・アーキテクチャが用いられている。Ｘ86アーキテクチャと呼ばれる一方は、Intel社によって開発されたものであり、あるファミリーのマイクロプロセッサ、特に80386、80486、Pentium（登録商標）およびＰ６（商標）をサポートする。PowerPC（登録商標）と呼ばれるもう一方は、International Business Machines社とMotorola社によって共同開発されたものであり、現在、複数のPowerPCプロセッサを含む（PowerPCはＩＢＭ社の商標である）。PowerPCプロセッサは縮小命令セット・コンピュータ（ＲＩＳＣ）プロセッサであり、一方、Ｘ86アーキテクチャは複合命令セット・コンピュータ（ＣＩＳＣ）アーキテクチャの一例である。

【０００３】Ｘ86アーキテクチャ用のソフトウェアの非常に大きなインストール済みのベースおよびPowerPCの処理能力を利用するために新たに書き込まれたソフトウェアを含む広い範囲のソフトウェアをサポートする必要がある。二つのアーキテクチャを具現化するプロセッサは、そのような具現化を達成して、いずれのアーキテクチャからのソフトウェアの実行をも可能にし、よってその潜在的な市場を拡大する。そのようなプロセッサが多重タスク環境においていずれのソフトウェア標準をも動的に作動させることができるならば、その有用性および潜在的な市場性は大幅に高まる。しかし、二つのアーキテクチャをこの方法で作動させることに関してはいくつか問題がある。これらのアーキテクチャは、二つのアーキテクチャに固有の異なる命令セットおよび異なるアドレス空間の使用を含む。ある対処法は、ＲＩＳＣプロセッサ上でＣＩＳＣアーキテクチャのソフトウェア・エミュレーションを実行するやり方である。しかし、この対処法は、ＲＩＳＣプロセッサの処理速度を犠牲にし、二つの異なるアーキテクチャ用に書かれた多重タスク処理ソフトウェアをサポートしない。求められるものは、ＲＩＳＣプロセッサ速度の潜在能力を最大に実現し、それでいて、二つのアーキテクチャ間の完全な互換性を保証するハードウェア的解決方法である。

【０００４】ＲＩＳＣプロセッサ、例えばPowerPCは、非常に限られた命令セットを有するが、最新のＣＩＳＣプロセッサは、簡約な命令および複雑な命令の両方を含む非常に広い命令セットを有している。ＣＩＳＣプロセッサの命令セットをＲＩＳＣプロセッサに加えるならば、ＲＩＳＣプロセッサの設計の他ならぬ目的を頓挫させることになろう。したがって、ＲＩＳＣプロセッサがその限られた命令セットを用いてＣＩＳＣ命令を実行することができるようにＣＩＳＣ命令セットを変換することは、はなはだしく大きな問題である。この問題は、適切なデコードを達成することができるよう、二つの異なる命令セット用に書かれた命令どうしを区別する必要性によってさらに悪化する。二つの異なるアーキテクチャ用に書かれたプログラムの多重タスク処理を達成しなければならないならば、それら二つの異なる命令セット用に書かれた命令の適切な識別およびデコードを動的かつユーザに対して透過的に行わなければならない。

【０００５】一つのプロセッサ上で二つのアーキテクチャをサポートし、いずれかのアーキテクチャ用に具現化されたソフトウェアの中で動的多重タスク処理を可能に

9

するには、そのプロセッサの実行装置が作動するところのアーキテクチャ・コンテキストに対して制御を加えることをまぬがれない。Ｘ86アーキテクチャとPowerPCのアーキテクチャとは、例えば、命令セットの設計においてだけでなく、各命令が実行リソース、例えばレジスタや結果フラグに対して配する仮説においても大きく異なる。これらのリソースはまた、プロセッサ中の相当な空き領域を占める。オペランドのサイズおよびタイプ、許容可能な演算ならびに演算の同期化要件もまた、アーキテクチャごとに異なる。

【０００６】PowerPCアーキテクチャは、固定小数点演算に用いられる32個の汎用レジスタ（ＧＰＲ）のセットと、浮動小数点演算に用いられる32個の浮動小数点レジスタ（ＦＰＲ）の別個のセットとを定義する。それぞれの固定小数点演算または浮動小数点演算には32個のレジスタのいずれを使用してもよい。レジスタ中に保持される値は常に右寄せされる。特定の命令がレジスタとメモリとの間でデータをロードし、記憶するために定義され、別個の命令セットがレジスタ中のデータに対して演算を加えるために定義されている。例えば、データをメモリからロードしたり、それをレジスタ中の値に加えるための命令は定義されていない。演算を実行するためには二つの別個の命令が要求されることになる。

【０００７】Ｘ86アーキテクチャは、８個のＧＰＲおよび８個のＦＰＲからなるセットを定義する。ＦＰＲは、レジスタ・ファイルとしてではなく、スタックとして編成されている。特定の命令が、ＧＰＲ中のレジスタを使用する方法に対して制限を加えて、ＧＰＲの汎用性を弱めている。例えば、ムーブ・ストリング（move string）演算は、インデックス・レジスタとしてのＥＤＩおよびＥＳＩの使用を制限する。ＧＰＲのうち４個においては、値を右寄せする必要はない。逆に、それらの値は、レジスタの第二のバイトから直接参照することができる。多くの命令が記憶場所に対して演算を実行することができる。例えば、加算命令は、一つのソースをＧＰＲから取り出し、別のソースをメモリから取り出し、結果を元のソース記憶場所に書き込むことができる。

【０００８】PowerPCの固定小数点および浮動小数点の実行命令はしばしば、三つのレジスタ・オペランド、すなわち二つのソース・オペランドおよび一つの目標を定義する。同様なＸ86命令は二つのオペランド、すなわちソース・オペランドおよびソース／目標オペランドだけを定義する。PowerPCプロセッサの命令に含まれるオペランドとは違い、Ｘ86オペランドの一方または両方が、レジスタだけではなく、記憶場所であることもできる。

【０００９】これら二つのアーキテクチャの実行リソース仮説の間には、それらをさらに区別する違いが他にも数多くある。例えば、PowerPCアーキテクチャは、一つのレジスタ内に８個の結果制御フィールドを定義し、比較および分岐の命令が８個のフィールドのいずれに対しても演算を加えることができ、コード生成において最適化コンパイラに大きな融通性を与える。Ｘ86アーキテクチャは、比較および分岐（飛越し）命令によって使用される１セットの結果制御しか定義しない。

【００１０】異なる命令セット・アーキテクチャに直接関連するさらなる問題は、命令そのものが主メモリから取り出され、プロセッサの取出しおよびデコード論理によってデコードされるところのアーキテクチャ・コンテキストに対する制御である。この場合もまた、Ｘ86アーキテクチャとPowerPCアーキテクチャとは、そのような論理に対して課す要件において大きく異なる。

【００１１】PowerPCアーキテクチャは、演算コードおよびオペランド情報が固定位置にある状態で、すべての命令をちょうど４バイトの長さに定義する。命令は常に１語（４バイト）境界上に配列され、その結果、キャッシュ、ページまたはセグメントの境界を越えることは決してない。

【００１２】他方、Ｘ86アーキテクチャは、オペランド情報が命令および演算コード前置に依存する不規則な位置にある状態で、可変長の命令を定義する。命令は、バイト境界にだけ配列され、したがって、キャッシュ、ページまたはセグメントの境界を越えることもある。その結果、アーキテクチャごとに命令取出しおよびデコード論理の設計に課される要求が大きく異なり、Ｘ86の要求がPowerPCの要求よりも効率的に具現化するのがはるかに困難である。

【００１３】もう一つの問題は、基礎にあるオペレーティング・システムが二つのアーキテクチャのアドレス変換機構の一方しか管理することができないことによる、メモリ管理である。これには二つの有意な理由がある。一つは、オペレーティング・システムの仮想メモリ管理部分（ＶＭＭと呼ぶ）が二つのアーキテクチャの一方のためにしか書かれていないことである。もう一つは、既存のプロセッサが、プロセッサの命令セット・アーキテクチャを直接サポートしているメモリ管理装置（ＭＭＵ）を１個しか含まないことである。ＭＭＵは、プロセッサ・ダイ上の有意な物理的空間を占め、そのため、物理的空間の制約が、二つのアーキテクチャをサポートする一つのプロセッサの具現化に対してさらなる妨げを課す。

【００１４】PowerPCおよびＸ86の両プロセッサ・アーキテクチャは、大きな仮想メモリ空間をより小さな物理アドレス空間の中にマッピングすることができるメモリ管理方式を定義する。いずれのアーキテクチャにおいても、仮想アドレスから物理アドレスへの変換は二段階処理である。まず、命令実行の一部として計算された実効アドレスがセグメント変換を受けて仮想アドレスを形成する。次に、この仮想アドレスがページング機構によって変換されて物理アドレスを形成する。これが基本的な処理であるが、言葉づかいが異なる場合もある。例え

ば、X86の文献は、実効アドレスを論理アドレスのオフセット部と呼び（セレクタが論理アドレスの残り部分を形成する）、仮想アドレスを線形（リニア）アドレスと呼ぶことがある。基本的なアドレス変換処理における類似性にもかかわらず、二つのアーキテクチャの間でセグメントおよびページ変換の詳細は大きく異なる。

【００１５】PowerPCアーキテクチャの64ビット版においては、ハッシュ合計されたセグメント・テーブル探索によって実効アドレス（ＥＡ）が仮想アドレス（ＶＡ）に変換される。ＥＡから抽出された実効セグメント識別（ＥＳＩＤ）の下寄り５ビットがハッシュ合計され、次いでアドレス空間レジスタと連結されて、メモリ中のセグメント・テーブル・グループの実アドレスを形成する。実効セグメントＩＤが元のＥＡのそれに一致する項目が見つかるまで個々のセグメント・テーブル・グループ項目が探索される。見つかると、仮想セグメントＩＤ（ＶＳＩＤ）がセグメント・テーブル・グループ項目から抽出され、元のＥＡのページおよびバイト・フィールドと連結されてＶＡを形成する。

【００１６】PowerPCアーキテクチャの32ビット版においては、ＥＡの上寄り４ビットが16個のセグメント・レジスタの１個への指標として使用される。ＶＳＩＤがセグメント・レジスタから抽出され、元のＥＡのページおよびバイト・フィールドと連結されてＶＡを形成する。

【００１７】X86アーキテクチャ・アドレス変換においては、直接セグメント・テーブル・ルックアップによってＥＡがＶＡに変換される。６個のレジスタ１個から取り出されたセレクタ値が二つの記述子テーブルの一方へのポインタとして使用される。セレクタによって指された記述子テーブル項目は基底アドレスを含み、この基底アドレスが元のＥＡに加えられてＶＡを形成する。X86の参照資料は通常、ＥＡを「オフセット」と呼び、セレクタとＥＡとの組合せを「論理アドレス」と呼び、ＶＡを「線形アドレス」と呼んでいる。

【００１８】ページ変換もまた、二つのアーキテクチャにおいて異なる。PowerPCアーキテクチャにおいては、ハッシュ合計されたページ・テーブル探索によってＶＡが物理アドレス（ＰＡ）に変換される。仮想セグメントＩＤの下寄り39ビットとＶＡからのページ・フィールドとがハッシュ合計され、次いで、ページ・テーブルの元のレジスタとマスク／併合されて、メモリ中のページ・テーブル・グループの実アドレスを形成する。仮想セグメントＩＤが元のＶＡのそれに一致する項目が見つかるまで個々のページ・テーブル・グループ項目が探索される。見つかると、実ページ番号がページ・テーブル・グループ項目から抽出され、元のＶＡのバイト・フィールドと連結されて64ビットＰＡを形成する。

【００１９】X86アーキテクチャにおいては、直接的な２レベルのページ・テーブル・ルックアップによってＶＡがＰＡに変換される。ＶＡの高位10ビットが、基底がページ・ディレクトリ・レジスタによって決定されるページ・ディレクトリ・テーブルへのポインタとして使用される。ＶＡによって参照されるページ・ディレクトリ・テーブルの中の項目は、ページ・テーブルの基底アドレスを含む。ＶＡの中間10ビットがこのページ・テーブルへのポインタとして使用される。このポインタによって参照されるページ・テーブル項目は、変換される仮想ページに対応する、メモリ中の物理ページの実ページ番号を含む。実ページ番号がＶＡのオフセット・フィールドと組み合わされて、最終的なＰＡを形成する。

【００２０】最後に、デュアルアーキテクチャ多重タスク処理プロセッサは、外部的および非同期的な割込みならびに同期的な例外、すなわち障害が受け入れられるところのコンテキストを管理することができなければならない。X86アーキテクチャとPowerPCアーキテクチャとは、割込みおよび例外に対して二つの異なる方式を定義している。

【００２１】PowerPCアーキテクチャは、外部的割込みの結果として制御が移される一つの場所を定義する。システムに問い合わせることによって割込みのベクトル番号が何であるかを決定する負担はソフトウェアに課される。すべての割込みおよび障害は、二つの可能な記憶場所の一方において、実アドレス・モードで受け入れられる。戻りアドレスは、割込み戻り命令による使用に備えてレジスタ中に記憶される。

【００２２】PowerPCに関するさらに詳細な情報については、Morgan Kaufman Publishing社（カリフォルニア州サンフランシスコ）出版の「PowerPC Architecture Specification for a New Family of RISC Processors」、IBM Corporation ICN 1-55860-C16-6（ＩＢＭ社、1993、1994）を参照されたい。

【００２３】

【発明が解決しようとする課題】したがって、本発明の目的は、一つの多重タスク処理オペレーティング・システムの下で二つの別個のアーキテクチャをサポートするプロセッサを提供することにある。

【００２４】本発明のもう一つの目的は、二つの別個のアーキテクチャそれぞれに対する拡張および／または限定（制限）を可能にし、プロセッサが作動するところのアーキテクチャ・コンテキストを制御する一つのマイクロプロセッサを提供することにある。

【００２５】本発明のもう一つの目的は、サポートされる二つのアーキテクチャのための命令セット定義を修飾することができ、それぞれのアーキテクチャにおいて定義されたリソースを他方のアーキテクチャ用に書かれたソフトウェアによってアクセスすることができるようにするマイクロプロセッサを提供することにある。

【００２６】本発明のさらなる目的は、実行リソースが作動するところのアーキテクチャ・コンテキストを決定して、実行コンテキストを一方のアーキテクチャからも

う一方のアーキテクチャに動的に切り換えることができるようにするマイクロプロセッサを提供することにある。

【００２７】本発明のさらに別の目的は、二つの別個のアーキテクチャをサポートし、ソフトウェアが種々の機構にアクセスして、マイクロプロセッサを、修飾モード制御を可能／不能にし、アーキテクチャ・コンテキスト制御機構に影響することができる既知の状態で初期化するモード制御装置を有するマイクロプロセッサを提供することにある。

【００２８】本発明のさらにもう一つの目的は、二つの別個のアーキテクチャをサポートし、両方のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行することができるメモリ管理ハードウェアを有し、アドレス変換を生じさせて、変換機構を一方のアーキテクチャからもう一方のアーキテクチャに切り換えることができるように設計されたマイクロプロセッサを提供することにある。

【００２９】本発明のさらに別の目的は、二つの別個のアーキテクチャをサポートし、両方のアーキテクチャについてメモリ保護検査を実行することができるメモリ保護検査ハードウェアを有し、一方のアーキテクチャのメモリ・リソースを他方のアーキテクチャのメモリ・リソースから保護することができるような方法でメモリ保護検査を生じさせるマイクロプロセッサを提供することにある。

【００３０】本発明のさらなる目的は、二つの別個のアーキテクチャをサポートし、割込みおよび例外が受け入れられるところのアーキテクチャ・コンテキストを決定して、割込みコンテキストを一方のアーキテクチャからもう一方のアーキテクチャに切り換えることができるようにするマイクロプロセッサを提供することにある。

【００３１】

【課題を解決するための手段】本発明によると、一つの多重タスク処理オペレーティング・システムの下で作動し、別個の命令セットおよびメモリ管理方式を有する第一および第二のアーキテクチャをサポートするマイクロプロセッサが提供される。このマイクロプロセッサは、第一のアーキテクチャの第一の命令セットの命令をデコードし、第二のアーキテクチャの第二の命令セットの命令をデコードする命令セット管理手段を含む。命令セット管理手段は、第一の命令セットのデコードされた命令を第二の命令セットの一つ以上の命令に対してマッピングする。マイクロプロセッサはさらに、前記第一および第二のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行するメモリ管理手段を含む。制御手段が、メモリから読み出されるプログラムのアーキテクチャ・コンテキストを、前記第一のアーキテクチャのコードまたは前記第二のアーキテクチャのコードのいずれかとして検出し、検出されたアーキテクチャ

・コンテキストに依存して、命令セット管理手段および前記メモリ管理手段を制御して、第一のアーキテクチャのアドレス変換と第二のアーキテクチャのアドレス変換との間で動的に切り換えて、第二のアーキテクチャの、一つ以上のマッピングされてデコードされた命令または直接デコードされた命令を実行する。

【００３２】本発明の具体的な実施態様においては、マイクロプロセッサには、二つのアーキテクチャに対する拡張および限定を可能にするアーキテクチャ・モード制御装置が設けられている。このような拡張および限定は以下を可能にする。

【００３３】・一方のアーキテクチャにおける既存の命令に対する新たな命令および拡張を可能にして、他方のアーキテクチャ独自のリソースへのアクセスを許容する。

【００３４】・一方のアーキテクチャが他方のアーキテクチャのリソースを十分に見ることができる。

【００３５】・一つのアドレス変換機構が一方のアーキテクチャのアドレスの変換をもう一方のアーキテクチャの変換に対してマッピングすることができるような効力をもつ。

【００３６】・一方のアーキテクチャの保護機構を他方のアーキテクチャのそれに対してマッピングする。

【００３７】・統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的割込みおよびページ変換ならびに保護関連の例外を一つの機構によって取り扱うことを許容する。

【００３８】加えて、アーキテクチャ・モード制御装置が、プロセッサの以下の分野を制御することにより、そのプロセッサが作動するところのアーキテクチャ・コンテキストを制御する（コンテキスト制御）。

【００３９】・二つのアーキテクチャ・モードによって共用される一つの命令取出し機構と、所与のコンテキスト制御にとって適当である場合にのみ活動するアーキテクチャごとに別々の命令デコード機構とがある。しかし、そのようなコンテキスト制御は、多数の命令取出し機構および／または一つの多重アーキテクチャ・デコーダを用いる具現化において使用することができる。

【００４０】・ すべての実行リソースが二つのアーキテクチャ・モードの間で共通である。しかし、そのようなコンテキスト制御は、共用または共通のリソースを用いない具現化において使用することができる。例えば、ある具現化は、X86アーキテクチャとPowerPCアーキテクチャとで別々のレジスタ・ファイルを有することができる。コンテキスト制御は、オペランドおよび結果アクセスに適当なレジスタ・ファイルを選択するために使用されるであろう。

【００４１】・サポートされる二つのアーキテクチャに共通のフォーマットを使用して、一つのメモリ管理装置

15

（ＭＭＵ）が具現化される。しかし、そのようなコンテキスト制御は、多数のＭＭＵを用いる具現化において使用されて、コンテキスト制御によって与えられるアーキテクチャに適当なＭＭＵによって変換を駆動することができる。スーパバイザレベルのコードをユーザレベルのコードから保護する場合にページ保護機構がプロセッサＭＭＵによって使用される。

【００４２】

【発明の実施の形態】一例として、ＩＢＭのPowerPCファミリーのプロセッサであるＲＩＳＣプロセッサを、IntelのＸ86ファミリーのプロセッサであるＣＩＳＣプロセッサのメモリ管理方式および命令セットをサポートするように変更する具体的な実施態様によって本発明を説明する。ただし、本発明は、他の異なるプロセッサにも応用しうることが理解されよう。そのうえ、本発明の教示は、それぞれが異なるメモリ管理方式および命令セットを有する二つのＲＩＳＣプロセッサ・アーキテクチャどうしの組合せまたは二つのＣＩＳＣプロセッサ・アーキテクチャどうしの組合せに応用してもよい。また、当業者であれば、本発明を、多数のプロセッサ・アーキテクチャをサポートするように拡張しうることが認められよう。

【００４３】ここで、図面、特に図１を参照すると、本発明を具現化しうるところの基本的なマイクロプロセッサ、例えばPowerPCマイクロプロセッサのブロック図が示されている。以下の論述は、マイクロプロセッサの基本的な動作を説明する。

【００４４】マイクロプロセッサ10は、そのシステム・インタフェース101を介して、64ビット・データ・バス121および32ビット・アドレス・バス122を含むシステム・バス12に接続されている。システム・バス12は、多様な入出力（Ｉ／Ｏ）アダプタおよびシステム・メモリ（図示せず）に接続されている。マイクロプロセッサ10は、とりわけ、システム・メモリに対して読み書きを実行するためにシステム・バス12を使用する。アドレスおよびデータ・バスのマスタシップのアービトレーションは、中央の外部アービタ（図示せず）によって実行される。

【００４５】システム・インタフェース101は、２要素読出し待ち行列1021および３要素書込み待ち行列1022からなるメモリ装置102に接続されている。読出し待ち行列1021は、読出し動作のためのアドレスを含み、書込み待ち行列1022は、書込み動作のためのアドレスおよびデータを含む。このメモリ装置102は、一方で、メモリ管理装置（ＭＭＵ）103に接続され、そこからアドレスを受ける。メモリ装置102はまた、命令およびデータの両方を記憶するキャッシュ104に接続されている。キャッシュ104中の命令およびデータ（オペランド）は、命令待ち行列1051と、プログラム・カウンタ1052と、送出し論理1053と、分岐履歴テーブル（ＢＨＴ）を有する分岐

16

予測装置（ＢＰＵ）とからなる命令装置105によってアクセスされる。

【００４６】送出し論理1053は、命令のタイプを決定し、それを、ここでは整数装置（ＩＵ）106および浮動小数点装置（ＦＰＵ）107によって表す複数の実行装置の対応する一つにディスパッチする。ＩＵ106は、スカラ（すなわち整数）演算を実行し、結果を汎用レジスタ（ＧＰＳ）ファイル1062に記憶する算術論理演算装置（ＡＬＵ）1061を含む。同様に、ＦＰＵ107は、浮動小数点演算を実行し、結果を浮動小数点レジスタ（ＦＰＲ）ファイル1072に記憶するＡＬＵ1071を含む。ＧＰＲファイル1062およびＦＰＲファイル1072それぞれからのデータ出力は、キャッシュ104に書き込まれ、そこからデータが、システム・メモリへの書込みのためにメモリ装置102に移される。データの計算に加えて、ＩＵ106はまた、命令装置105によるアクセスのためのアドレスを計算し、一時的にこれらのアドレスをレジスタ1063中に記憶する。レジスタ1063中のアドレスが、ＢＰＵ1054によって出力されるアドレスとともに、ＭＭＵ103に供給される。

【００４７】命令装置105はまた、割込み（プロセッサに対して外部にあるハードウェアによって開始される非同期的事象）ならびに例外および障害（命令を取り出したり、デコードしたり、実行したりする結果として生じる同期的事象）を処理する。割込みは、システム・インタフェース101を介してマイクロプロセッサ10に送られ、送出し論理1053に転送される。例外および障害（以下、単に「例外」と呼ぶ）は、命令待ち行列1051、メモリ管理装置103、ＩＵ106またはＦＰＵ107のいずれによっても検出し、送出し論理1053に転送することができる。送出し論理1053は、各例外の優先順位を定め、それらを、命令境界上の分岐予測装置1054に送る。そして、分岐予測装置1054が、命令装置105が命令を取り出す場所を適当な割込み／例外ハンドラの場所に変更する。

【００４８】命令およびオペランドは、システムメモリからキャッシュ104を介して自動的に取り出されて命令装置105に入れられ、そこで、１クロックあたり命令３個の最大速度で実行装置にディスパッチされる。ロードおよび記憶の命令は、整数および浮動小数点レジスタ・ファイルとメモリ・システムとの間を行き来するオペランドの動きを指定する。命令またはデータアクセスが行われるとき、命令装置105（命令アクセスの場合）または整数装置106（データアクセスの場合）によって論理アドレス（実効アドレス）が計算される。メモリ管理装置103が実効アドレスを物理アドレスに変換し、それをキャッシュ104に転送する。メモリ管理装置103はまた、実効アドレスを物理アドレスに変換する際、マイクロプロセッサ10の現在の特権を検査して、メモリにアクセスしてもよいことを検証する。物理アドレス・ビットの一部がキャッシュ・タグ・ビットＩＮ104と比較されて、

キャッシュ・ヒットが生じたかどうかを判定する。キャッシュ104においてアクセスを逸するならば、物理アドレスを使用してシステム・メモリにアクセスする。

【００４９】マイクロプロセッサ10は、ロード、記憶および命令取出しに加えて、テーブル探索のための他の読み書き演算、キャッシュ・ミスののち、もっとも以前に使用された（ＬＲＵ）セクタがメモリに書き込まれる場合のキャッシュ・キャストアウト演算および変更されたセクタが別のバス・マスタからのスヌープ・ヒットを経験する場合のキャッシュセクタ・スヌープ・プッシュアウト演算を実行する。すべての読み書き演算はメモリ装置102によって取り扱われる。干渉性を維持するため、書き込み待ち行列1022がスヌーピングに含まれる。メモリは、デバイスがバス・マスタシップと求めて競合することを許すアービトレーション機構を介してアクセスされる。

【００５０】マイクロプロセッサ10はまた、種々の装置が作動するところのアーキテクチャ・コンテキストならびに所与のアーキテクチャ・コンテキストの下でそれらの装置に加えられるかもしれないアーキテクチャ修飾または拡張を制御するモード制御装置108を含む。メモリ管理装置（ＭＭＵ）103が、どのアーキテクチャに命令が適合するのかを検出する。例えば、ＭＭＵ103は、命令がＸ86命令であるのか、PowerPC命令であるのかを判断し、ページ・モード制御（Ｐ）ビットにより、相応なアーキテクチャ・コンテキストをモード制御装置108に知らせてもよい。モード制御装置108はまた、PowerPC機械状態レジスタ命令再配置（ＭＳＲ　ＩＲ）ビット109から仮想（Ｖ）ビットを受ける。モード制御装置108は、修飾（Ｑ）ビットを生成し、保持する。このＱビットが命令装置105に出力され、命令装置105が命令を取り出し、デコードする方法を決定する。Ｑビットはまた、整数装置（ＩＵ）106および浮動小数点装置（ＦＰＵ）107に出力され、どのレジスタ・リソースが整数装置106および浮動小数点装置107に利用できるかを決定する。モード制御装置108からのＱビットはまた、割込み／例外が生じた際に分岐予測装置1054が命令装置105に指示し直す方法を統制する。モード制御装置はまた、ＰおよびＶビットならびにモード制御装置108によって内部的に生成されるＱビットに基づき、現在一方のアーキテクチャで作動中のプログラム中の点を他方のアーキテクチャで作動させるための異なるプログラムへのモード切換えが行われたときに制御するモード切換え信号を生成する。

【００５１】アーキテクチャへの拡張は、新たな命令およびレジスタを含むかもしれない。限定は、特定のレジスタへのアクセスを不能にすること、すなわち、アドレスを変換する方法に対する制限を含むかもしれない。拡張および／または限定の背後にある第一の目的は、それぞれのアーキテクチャで書かれたソフトウェアが、他方

のアーキテクチャによって定義されたリソースに対してもある程度のアクセスを有することを許すことである。第二の目的は、アーキテクチャ・コンテキストが変わるとき、それぞれのアーキテクチャで書かれたソフトウェアが、他方のアーキテクチャで書かれたソフトウェアに制御を移すことを許すことである。実際には、アーキテクチャ・モード制御装置108によって可能になる拡張は、ソフトウェアがアーキテクチャ・コンテキストにおける変更を呼び出すことを許す機構を含む。

【００５２】プロセッサのアーキテクチャ・コンテキストは、単に、そのプロセッサが作動するところの現在の命令セット・アーキテクチャ・コンテキストである。例えばPowerPCおよびＸ86の両アーキテクチャをサポートするプロセッサにおいては、コンテキスト制御が、プロセッサがPowerPCプロセッサのように挙動するのかＸ86プロセッサのように挙動するのかを決定する。アーキテクチャ修飾制御（拡張／限定を可能にする）およびアーキテクチャ・コンテキスト制御（コンテキストを決定する）はいずれも以下のそれぞれに直接的な影響を及ぼすことができる。

【００５３】・命令セット定義機構
・命令コード化機構
・命令演算コード長さ機構
・アドレス変換機構
・セグメントおよびページ・テーブル編成機構
・保護機構
・割込みアーキテクチャ機構
・メモリ・アドレス指定性機構
・レジスタ・セットおよびレジスタ機構
・条件、フィールドおよび結果機構

【００５４】本発明の好ましい実施態様においては、モード制御装置108は、一方のアーキテクチャのアドレス変換を他方のそれに対してマッピングすることができる一つのＭＭＵ103によって制御される。モード制御装置108はさらに、メモリ保護検査ハードウェアを制御して、両方のアーキテクチャ用に書かれたソフトウェアを一つのオペレーティング・システムの制御の下で保護し、多重タスク処理することができるようにする。

【００５５】図２は、PowerPCアーキテクチャのうち、実効アドレスから仮想アドレスへの変換（セグメント化）を実行する部分のブロック図である。64ビットの実効アドレスがレジスタ21中に保持され、一方で、アドレス空間レジスタ（ＡＳＲ）22がセグメント・テーブルの実アドレスを保持する。ＥＡレジスタ11から抽出された実効セグメント識別（ＥＳＩＤ）の下寄り５ビット、すなわちビット31～35がハッシュ機能23によってハッシュ合計され、次いでＡＳＲ22のビット０～51と連結されて、セグメント・テーブル項目レジスタ24中に実アドレスを形成する。この実アドレスの最後のバイトは０に強要される。セグメント・テーブル項目レジスタ24は、40

19

96バイトを含むメモリ中にセグメント・テーブル25をアドレス指定する。ハッシュ合計されたセグメント・テーブル探索によって実効アドレス（ＥＡ）が仮想アドレス（ＶＡ）に変換される。実効セグメントＩＤが元のＥＡのそれに一致する項目が見つかるまで個々のセグメント・テーブル・グループ項目が探索される。見つかると、セグメント・テーブル・グループ項目26から仮想セグメントＩＤが抽出され、元のＥＡのページおよびバイト・フィールドと連結されて、80ビットＶＡをレジスタ27中に形成する。

【００５６】図３に示すＸ86アーキテクチャ・アドレス変換においては、直接的なセグメント・テーブル・ルックアップによってＥＡがＶＡに変換される。６個のレジスタ31の１個から取り出されたセレクタ値が二つの記述子テーブル32の一方へのポインタとして使用される。セレクタによって指された記述子テーブル項目は、記述子テーブル基底レジスタ33から読み出され、加算器35によってＥＡレジスタ34中の元のＥＡに加えられてＶＡをＶＡレジスタ36中に形成する基底アドレスを含む。Ｘ86の参照資料は通常、ＥＡを「オフセット」と呼び、セレクタとＥＡとの組合せを「論理アドレス」と呼び、ＶＡを「線形アドレス」と呼んでいる。

【００５７】ページ変換もまた、二つのアーキテクチャ間で異なる。図４に示すPowerPCアーキテクチャにおいては、ハッシュ合計されたページ・テーブル探索によってレジスタ27中のＶＡが物理アドレス（ＰＡ）に変換される。具体的には、レジスタ27中の80ビット仮想アドレスのビット52～67（ページ・フィールド）がレジスタ42中の23個の０と連結される。そして、レジスタ27中の仮想セグメントＩＤのビット13～51とレジスタ42の内容とがハッシュ機能43においてハッシュ合計されて、レジスタ44中に39個のビットを生成する。ハッシュ・テーブル・レジスタ45中のビット58～63がデコーダ46によってデコードされて、レジスタ47中に28ビットのマスクを生成する。レジスタ44のビット０～27が、ＡＮＤゲート48の中でレジスタ中のマスクによってマスクされ、マスクされた出力がＯＲゲート49の中でレジスタ45のビット18～45と合わされて、ページ・テーブル起点レジスタ50中の中間の28ビットを形成する。レジスタ50の最初の18ビットがレジスタ45のビット０～17から直接読み出され、次に高い１１ビットがレジスタ44のビット28～38から読み出される。レジスタ50のもっとも高い７ビットが０に強要されて、ページ・テーブル51のメモリ中の実アドレスを形成する。仮想セグメントＩＤが元のＶＡのそれに一致する項目52が見つかるまで個々のページ・テーブル・グループ項目が探索される。見つかると、ページ・テーブル・グループ項目52から実ページ番号が抽出され、ＶＡレジスタ27の元のＶＡのバイト・フィールド、すなわちビット68～79と連結されて、物理アドレス・レジスタ53中に64ビットのＰＡを形成する。

20

【００５８】図５に示すＸ86アーキテクチャにおいては、直接的な２レベル・ページ・テーブル・ルックアップによってＶＡがＰＡに変換される。より詳細には、ＶＡレジスタ36中のＶＡの高位10ビットが、基底がページ・ディレクトリ・レジスタ55によって決定されるページ・ディレクトリ・テーブル54へのポインタとして使用される。ＶＡによって参照されるページ・ディレクトリ・テーブル54中の項目は、ページ・テーブル56の基底アドレスを含む。レジスタ36中のＶＡの中間10ビットがこのページ・テーブル56へのポインタとして使用される。このポインタによって参照されたページ・テーブル項目は、変換される仮想ページに対応するメモリ中の物理ページの実ページ番号を含む。この実ページ番号が加算器57によってレジスタ36中のＶＡのオフセット・フィールドと合わされて、物理アドレス・レジスタ５８中の最終的なＰＡを形成する。

【００５９】一つのプロセッサ上で作動することができる二つのアーキテクチャの具現化を達成するためには、図１に示すモード制御装置108によって制御しなければならない必須要素がいくつかある。具体的には、モード制御装置108は、以下の点で異なる二つのアーキテクチャを有する場合を取り扱う。

【００６０】・命令セット定義、コード化および演算コード長
・アドレス変換
・セグメントおよびページ・テーブル編成
・保護機構
・割込みアーキテクチャ
・メモリアドレス指定性
・レジスタ・セット
・条件、フィールドおよび結果機構

【００６１】アーキテクチャどうしが異なるすべての区域において、プロセッサの具現化は、各アーキテクチャの異なる側面を個々にサポートするハードウェアを含んでもよいし、すべてのアーキテクチャをサポートすることができる共通のハードウェアリソースを含んで、ハードウェアリソースの冗長性を除いてもよい。別個のハードウェア要素を使用してアーキテクチャをサポートする状況においては、アーキテクチャ・モード制御装置108が、適当なハードウェア要素を可能にし、他の要素を不能にする責任を負う。例えば、プロセッサが、サポートされるアーキテクチャのそれぞれについて別々の命令セット・デコーダを具現化するならば、アーキテクチャ・モード制御装置は、現在使用中のアーキテクチャのための命令セット・デコーダを可能にし、残りのアーキテクチャのためのデコーダを不能にする。共通のハードウェアリソースを用いて冗長性を除く状況においては、アーキテクチャ・モード制御装置は、そのようなハードウェアに対し、適当ならば、現在のアーキテクチャ・コンテキストの規則の下で、そのコンテキストのために定義さ

れた拡張を利用しながら作動するよう指示する。例えば、プロセッサが、物理的ハードウェアおよび配線を最小限にするために共通のレジスタ・ファイルを具現化するならば、アーキテクチャ・モード制御装置は、所与のハードウェア・コンテキストの下で共通レジスタ・ファイル中のどのレジスタにアクセスしてもよいかを制御する。

【００６２】したがって、本発明によると、モード制御装置108は、一つのアーキテクチャの異なる側面を個々にサポートするハードウェア・リソースの間で選択を行う。制御装置108はまた、多数のアーキテクチャの異なる要素をサポートすることができるような共通のハードウェア・リソースをも制御する。そのようなハードウェア・リソースは、冗長ハードウェアを除いて、それにより、物理的空間、電力消費を節約し、プロセッサ性能を改善し、全体の設計を簡素化するために具現化される。

【００６３】アーキテクチャ修飾制御機構は、モード制御装置108中のプロセッサ機能制御レジスタ（図６に示す）中に保持されるビットの値によって決定される。このビットは、いずれかのアーキテクチャで作動中のソフトウェアによって設定することができる。このビットが０であるとき、アーキテクチャ、例えばPowerPCアーキテクチャまたはＸ86アーキテクチャのいずれにも修飾が加えられない。すなわち、これが正常な作動状態である。しかし、このビットが１であるとき、アーキテクチャ・コンテキスト制御機構の制御の下で現在作動中のアーキテクチャを修飾する。より詳細には、このビットが１であるとき、マイクロプロセッサは、マイクロプロセッサが、現在のアーキテクチャによって定義されない、他方のアーキテクチャ中のレジスタに対して読み書きすることを許す、利用できる新たな命令を現在のアーキテクチャ中に有する。

【００６４】一例として、Ｘ86アーキテクチャは、PowerPCアーキテクチャには存在しない、ＦＬＡＧＳと呼ばれる状態／制御レジスタを有している。そのため、マイクロプロセッサがＦＬＡＧＳレジスタに対して読み書きすることを許す新たなPowerPC命令が定義され、修飾制御ビットが１である場合にのみ、この命令を使用することができる。同様に、このビットは、現在のアーキテクチャ中のいくつかの命令を不能にする。一例として、修飾制御信号が１である場合、PowerPCアーキテクチャは、アーキテクチャ・コンテキストがＸ86モードであるとき、Ｘ86アーキテクチャのＩＮＶＬＰＧ（ＴＬＢ中のページ・テーブル項目を無効化）命令をもはや認識しない。修飾制御機構が１であり、コンテキスト制御機構が０であるとき（PowerPCアーキテクチャ・モード）、Ｘ86アーキテクチャ型リソースへのアクセスをPowerPCアーキテクチャに与える、PowerPCアーキテクチャに対する拡張が可能になる。

【００６５】例えば、PowerPCアーキテクチャの命令取

出し機構に対して、PowerPCプロセッサ用に設計されたソフトウェアがバイト整列されたＸ86ソフトウェアへと分岐することを許す機能強化が成される。PowerPCプロセッサ用に設計されたソフトウェアが64ビットＸ86記述子レジスタに対して読み書きすることを許すさらなるPowerPC命令が可能になる。最後に、オペレーティング・システムが、PowerPCプロセッサ用に設計されたコードを含む記憶場所を、Ｘ86コードを含む記憶場所から区別することを許す、PowerPCプロセッサ・ページング機構に対する拡張が可能になる。

【００６６】図６は、例示的なモード制御装置108の関連の論理を示す論理図である。ハードウェア機能制御レジスタ61は、上記に説明したように、ソフトウェアによってセットされるラッチである。ラッチがセットされると、ラッチはＡＮＤゲート６２を可能にし、このゲートがモード切換え信号を出力する。モード切換え信号は、処理されている現在のコードが完了したところでアーキテクチャ・コンテキスト切換えを可能にする信号である。モード切換え信号は、Ｖビット、ＰビットおよびＱビットの状態に依存して生成される。Ｖ（仮想モード）ビットは、PowerPC機械状態レジスタ命令再配置（ＭＳＲ　ＩＲ）ビット109（図１を参照）によってモード制御装置108に入力され、PowerPC命令アドレス変換が可能にされた状態（Ｖ＝１）か、不能にされた状態（Ｖ＝０）かを示す。Ｐ（ページ・モード制御）ビットは、ＭＭＵ103によって供給され、現在のページがPowerPCページである（Ｐ＝０）のか、Ｘ86ページである（Ｐ＝１）のかを示す。Ｑビットはアーキテクチャ修飾ビットである。このビットは、図１に示すように、命令装置105、整数装置（ＩＵ）106および浮動小数点装置（ＦＰＵ）107に供給される。ＰビットおよびＶビットがＡＮＤゲート63に入力され、このゲートの出力が排他的ＯＲ（ＸＯＲ）ゲート64の一方の入力に供給される。ＸＯＲゲート64の第二の入力は、Ｑビットを記憶するレジスタ66に接続されたインバータ65によって供給を受ける。レジスタ66の出力は、マルチプレクサ（ＭＵＸ）67の一方の入力に直接供給され、また、インバータ68を介して、ＭＵＸ67の他方の入力にも供給される。ＭＵＸ67は、図９を参照しながらさらに詳細に示し、説明するように、分岐予測装置（ＢＰＵ）1054（図１）からの遅延モード切換え信号（記号「′」によって示す）によって制御される。

【００６７】初期値（すなわちハードウェア・リセット時の値）は、Ｖ＝０、Ｐ＝Ｘ、Ｑ＝０およびレジスタ61＝０である。このとき、Ｘは「don't care（不問）」を意味する。モード切換え信号は、レジスタ61を１に設定することによって可能になる。Ｑビットが０であり、ハードウェア機能制御レジスタ61が同様に１（Ｘ86アーキテクチャ・モード）に設定されると、Ｘ86アドレス変換機構に特定の限定が加えられる。具体的には、Ｘ86ページング機能が不能になり、それに代えて、セグメントお

よびページ変換の完全なPowerPC アドレス変換機構が配される。Ｘ86セグメント変換機構はなおも利用される。

【００６８】特定の他のページング関連の動作もまた不能になる。具体的には、ページ・ディレクトリ基底レジスタに対するＸ86書き込みは効果を有さず、Ｘ86ページ変換モードを可能にすることができず、Ｘ86ソフトウェアは、図１のＭＭＵ103によって内部的に保持された項目を無効化することができない。代わりに、これらの動作はプロセッサによって内部的に捕らえられ、適切なものとして取り扱われる。Ｘ86アーキテクチャに対する拡張もまた、アーキテクチャ修飾機構状態ビットによって可能になる。具体的には、Ｘ86命令は、特定のPowerPCアーキテクチャ型レジスタにアクセスすることができる。

【００６９】マイクロプロセッサにおけるアーキテクチャ・コンテキスト制御は、モード制御装置108によって生成され、レジスタ66中に保持されるＱビットの値によって決定される。例えば、このビットが０であるとき、マイクロプロセッサは、実行のための命令取出しおよびデコードから割込みおよび例外の取扱いのためのアドレス変換まで、PowerPCアーキテクチャに関連する完全な規則のセットに従う。このビットが１であるとき、プロセッサは完全なＸ86アーキテクチャ規則に従う。好ましい実施態様においては、リセット後のこのビットの初期値は０であり、プロセッサを、完全なＸ86アーキテクチャに従うところのモードに入れる。

【００７０】Ｑビットの値をアーキテクチャ・モード制御装置108によって変更して、マイクロプロセッサを、Ｘ86モードで作動させながらもPowerPC モードに入れたり、PowerPCモードで作動させながらもＸ86モードに入れたりすることができる。これはＭＵＸ67によって達成される。図７は、好ましい実施態様のモード制御装置108がアーキテクチャ・コンテキスト制御信号の値を変更する方法を示す状態図である。図７は、四つの状態、すなわち、完全なＸ86アーキテクチャ・モード、修飾Ｘ86アーキテクチャ・モード、修飾PowerPCアーキテクチャ・モードおよび完全なPowerPCアーキテクチャ・モードを示す。好ましい実施態様においては、モード制御装置108は、Ｖ、ＰおよびＱビットの値を使用して、プロセッサを一つのアーキテクチャから別のアーキテクチャに進める。上述したように、Ｑビットはレジスタ66中に保持されており、二つのアーキテクチャのいずれかで作動中のソフトウェアによって設定することができる。ビットＶもまた、ソフトウェア制御の下にあり、例外時にクリアされる。ビットＰはＭＭＵ103によってモード制御装置108に供給される。

【００７１】好ましい実施態様においては、アーキテクチャ・コンテキストのいずれかからのマイクロプロセッサのリセットがプロセッサを完全なＸ86モードに入れる。すなわち、マイクロプロセッサがリセットされるならば、コンテキストおよび修飾制御機構が何らかの既知の初期状態に置かれる。好ましい実施態様においては、リセットはプロセッサを修飾なしでＸ86コンテキストに残す。Ｑ（図７を参照）が１にセットされない限り、プロセッサが完全なＸ86モードを出ることはない。しかし、Ｖ、ＰおよびＱがすべての１にセットされるならば、マイクロプロセッサは修飾Ｘ86モードに入り、Ｖ、ＰおよびＱが１であり続ける限り、そのモードにとどまる。修飾PowerPCモードには、完全なＸ86モードまたは修飾Ｘ86モードのいずれかから、二つの方法のいずれかを用いて、すなわち、Ｑが１であるときにＶを０にセットすることにより、または、Ｐを０にセットしながらＶおよびＱを１にセットすることにより、入ることができる。完全なPowerPCモードには、Ｑが０にセットされるならば、両方の修飾モードのいずれからでも入ることができる。

【００７２】図８は、送出し論理1053（図１）のデータの流れを示して、マイクロプロセッサの命令セット管理動作を説明する高レベル機能ブロック図である。これは、以下さらに詳述する図９にさらに詳細に示すハードウェアのデータの流れである。起動またはリセット時に、マイクロプロセッサは、省略時モードをとることによって始まる初期化モードに入る。好ましい実施態様においては、省略時モードは、図７に示すような、モード制御装置108に配線接続された完全なＸ86モードである。ＭＭＵ103は初めクリア状態である。起動またはリセット時に、命令コードが主メモリからシステム・インタフェース101およびバス12を介して検索される。このコードが上述の通常データ経路を介して命令装置105に供給される。

【００７３】命令コードは、図８に示すように、簡単なPowerPC命令を取り扱うデコード機能ブロック70、簡単なＸ86命令を取り扱うデコード機能ブロック71、複雑なPowerPC命令を取り扱うデコード機能ブロック72および複雑なＸ86命令を取り扱うデコード機能ブロック73に対して並列に供給される。ここに説明する例において「簡単な」命令とは、基本的な演算クラスに対してマッピングし、一つの実行装置によって取り扱うことができる命令をいう。例えば、ロード演算、記憶演算および単一算術演算はすべて簡単な命令である。この定義により、他すべての命令は「複雑」である。例えば、Ｘ86のリピート・ムーブ・ストリング（repeat movestring）またはPowerPCのロード・マルチプル・ワード（load multiple word）は、複雑な命令の例である。デコード機能70の出力、すなわちデコードされた簡単なPowerPC命令は、マルチプレクサ（ＭＵＸ）74の一方の入力に供給される。デコード機能71の出力、すなわちデコードされた簡単なＸ86命令は、まず、変換装置75（例えば、テーブル・ルックアップＲＯＭ）に供給され、この変換装置が、対応する簡単なPowerPC命令をＭＵＸ74の第二の入力に出力

25

する。デコード機能72および73の出力は、マルチプレクサ（ＭＵＸ）76に供給され、このマルチプレクサの出力がマイクロコード読出し専用メモリ（ＲＯＭ）77に供給される。

【００７４】ＲＯＭ77は、PowerPCおよびＸ86の両方の複雑なデコードされた命令のための命令のテーブルを有している。複雑なPowerPCのデコードされた命令の場合、ＲＯＭ77は、多数の、簡単なPowerPCデコードされた命令を出力する。ＭＵＸ76から出力される複雑なデコードされたＸ86命令が、ＲＯＭ77中の対応する多数の簡単なPowerPC命令に対してマッピングされる。このように、ＭＵＸ74およびＲＯＭ77の出力は、一つ以上の簡単なPowerPC命令である。モード制御装置108（図１および６）からのＱビットがマルチプレクサ74および76を制御して、デコード機能70、72および71、73によってそれぞれ出力されるPowerPCまたはＸ86のデコードされた命令のいずれかを選択させる。ＭＵＸ74およびＲＯＭ77の出力は、ＯＲゲート79によって制御される第三のＭＵＸ78に供給される。ＯＲゲート79は、「複雑な」命令がデコード機能72または73の一方によって検出された場合、そのデコード機能から「有効」出力を受ける。図示してはいないが、これらの「有効」信号はまた、Ｑビットによって修飾される。このように、ＯＲゲートの出力は、複雑な命令と簡単な命令との間で、すなわち、ＲＯＭ77の出力とＭＵＸ74の出力との間で選択を行う。

【００７５】Ｘ86およびPowerPCの修飾モード（図７を参照）を準備するため、初期化ソフトウェアは以下の段階を実行する。

【００７６】・省略時Ｘ86モードからPowerPCモードへの即時コンテキスト切換えを実行するため、ハードウェア機能制御レジスタを１にセットする。

【００７７】・PowerPC仮想環境を初期化する（すなわち、外部メモリ中に必要なページ・テーブルを準備し、ＢＡＴおよびセグメント・レジスタを初期化する）。

【００７８】・機械状態レジスタ（ＭＳＲ）中の命令再配置（ＩＲ）ビットを１にセットすることにより、命令再配置を可能にして、ひいては命令アドレスの仮想変換を可能にする。

【００７９】・アプリケーション起動を取り扱う適当なソフトウェア機構（例えばＸ86またはPowerPCコード）に対して分岐命令を実行する。

【００８０】ハードウェア・リセットののち、プロセッサは、Ｘ86モードで実行する（省略時）。したがって、Ｑビットがモード制御装置108によって０にセットされ、ハードウェア機能制御レジスタもまた０に初期化される。初期化ソフトウェアは、続いて、コンテキスト切換えを可能にするため、ハードウェア機構制御レジスタを１にセットすべきである。命令再配置はオフである（すなわちＶビットが０である）ため、これはまた、モード制御装置108に対し、Ｘ86コードの実行からPowerPCコードの実行へのコンテキスト切換えを実行するよう強要する。機能制御レジスタ書込みに続く次の命令は、PowerPC命令として実行される。

26

【００８１】次に、初期化プロセスにおいて、ページ・テーブル52（図４）が準備され、命令再配置が可能になったならば（すなわち、ＭＳＲ　ＩＲビットが１にセットされ、その結果、Ｖビットが１にセットされたならば）実効状態になる、修飾PowerPCモード（図７）を可能にするためのエントリが行われる。これもまた、PowerPCセグメントレジスタおよびＢＡＴレジスタをそれらの適当な値に初期化することを要する。そして、Ｘ86記述子テーブルを、Ｘ86アプリケーションによる使用に備えて初期化することになる。最後に、PowerPCのＭＳＲレジスタ中のＩＲビットを１にセットし、それにより、Ｖビットを１にセットすることによって命令再配置が可能になるであろう。最後の命令は、最初のプログラムを開始する記憶場所への分岐命令となる。目標コードは、Ｘ86コードまたはPowerPCコードのいずれかであることができる。いずれであれ、そのコードは４KBブロックにおいて管理される。

【００８２】分岐命令が実行すると、ＱおよびＶの値は、Ｑ＝１（PowerPCコードを実行）およびＶ＝１（命令再配置／仮想変換可能）になる。したがって、Ｑビットの次の状態が、モード制御装置108により、分岐命令の目標に対応するページ・テーブル項目からアクセスされたＰビットの値から決定される。Ｐビットの値は、ＭＭＵ103からモード制御装置108に提供されるものである。この時点で、ソフトウェア初期化が完了する。

【００８３】初期化が完了したならば、ＭＭＵ103がＴＬＢルックアップを実行する。ルックアップの結果、ヒットし、Ｐビットが１であるならば、ＭＭＵ103は、そのコードがＸ86コードであることを認知し、モード制御装置108に知らせ、ページ・モード制御ビットＰによって状態を戻して、それにより、コンテキスト・モードに関してモード制御装置に指示を出す。この流れから理解されることは、入ってきた命令がＭＭＵ103中に位置するか、ＭＭＵに加えられるとき、その命令がＭＭＵ103をしてモード制御装置108のモードを切り換えさせるということである。換言するならば、ＭＭＵ103のアドレス指定方式がモード制御装置108を駆動してモードを変更させるのである。しかし、入ってきた命令ＴＬＢ中にない（ミス）ならば、ページ・テーブル・ウォークを実行して主メモリ・ルックアップ・テーブル中にその命令を探す。そこにもないならば、特殊な割込みが実施され、取り扱われる。

【００８４】命令がページ・テーブル中にあると仮定すると、その命令はＭＭＵ103にロードされ、Ｐビットが現在のコンテキスト・モードと異なるならば、上記のようにＭＭＵ103はモード制御装置108のモードを切り換える。ＭＭＵ103は常にモード制御装置のコンテキスト・

モードを打ち消す。この命令が次に実行するべき命令であるならば、モードの切換えが起こる。修飾PowerPCモードから修飾Ｘ86モードへの切換えは（図７）は、Ｐビット（Ｐ＝１）によってモード制御装置108に知らせることによって実施される。そこで、モード制御装置108が、整数装置（ＩＵ）106および浮動小数点装置（ＦＰＵ）107に対し、すべての命令をＸ86命令として解釈するように伝える。さらに詳細には、アドレス変換機構ならびにセグメントおよびページ・テーブル編成機構がＩＵ106中に存在する。モード制御装置108が、レジスタが何個あり、それらのレジスタがどのフォーマットを使用しているのかをＩＵ106に伝える。モード制御装置108は、ＦＰＵ107に対し、80ビットＸ86スタイルを使用すべきか、64ビットPowerPCスタイルを使用すべきかを伝え、それにより、同じレジスタ・ファイルの精度を制御する。

【００８５】図８は、命令の流れが命令装置に進入し、それがデコードされる方法を示す。モード制御装置が送出し論理によってデコードされた命令の選択を制御する。図９は、図８の機能ブロック図に示す機能を実行する命令装置105の詳細なハードウェア・ブロック図である。図１の命令待ち行列1051は、図９に示す取出し整列装置801および命令待ち行列802を含む。取出し整列装置801は、命令取出しバスに接続され、入力として、命令キャッシュから取り込まれた命令および現在の命令アドレスを有している。取出し整列の要件はPowerPCアーキテクチャとＸ86アーキテクチャとで異なるため、モード制御装置108からのＱビットを使用して、命令が、命令待ち行列802に移される前に、いかにして取出し整列ハードウェア801によって整列されるかを決定する。例えば、Ｘ86命令は、１～15個のバイトを含み、いかなるバイト境界上に配列されることもできるが、PowerPC命令は、常に長さ４バイトであり、したがって、常に４バイト境界上に配列される。取出し整列ハードウェア801は、命令取出しバスからの命令を、現在の命令の長さおよびアドレスに基づく適当な量だけシフトすることができなければならない。加えて、Ｘ86命令は長さが異なり、命令取出しバスのどこにでも配列されるため、Ｘ86命令は、一つの命令キャッシュ項目をあふれさせ、次の項目に入り込むおそれがある。アーキテクチャ・モード制御信号を使用して、取出し配列ハードウェア801が、連続する命令キャッシュ項目からの命令を併合し、命令待ち行列802の中に整列させるようにすることができる。

【００８６】命令は、命令待ち行列802からＸ86命令デコーダ803およびPowerPC命令デコーダ804に転送される。Ｘ86命令デコーダ803は、図８のデコード機能71および73を実行し、PowerPCデコーダ804は、同図のデコード機能70および72を実行する。Ｑビットを使用して、図８を参照しながら説明したように、望みのアーキテクチャ・コンテキストに依存してＸ86命令デコーダ803またはPowerPC命令デコーダ804のいずれかを可能にする。これらのデコーダを使用して、すべての命令を、マイクロプロセッサによって実行がサポートされるところの共通の命令セットに変換する。例えば、Ｘ86の「ADD EAX,EBX」命令およびPowerPCの「add r8,r8,r11」命令は、いずれもそれぞれのデコーダにより、共通の「add r8,r8,r11」命令に変換される。より複雑な命令は、単一のより複雑な命令の機能を実行する簡単な一連の命令を生成するそれぞれのデコーダ803または804によって取り扱われる。例えば、PowerPC命令デコーダ804は、PowerPCのロード・マルチプル・ワード命令を個々のロードの連続としてエミュレートする。

【００８７】モード制御装置108からのＱビットはまた、図９を参照しながら説明したように、デコーダ803および804に対して直接的な効果を及ぼす。具体的には、PowerPCアーキテクチャにおいて、PowerPCソフトウェアがＸ86浮動小数点環境を管理し、Ｘ86記述子レジスタに対して読み書きすることを許す新たな命令が可能にされる。Ｘ86アーキテクチャにおいては、Ｑビットが、ページ変換を可能にし、ページ・ディレクトリ基底レジスタを初期化し、ＴＬＢ中の項目を無効化するＸ86命令を不能にする。Ｑビットはまた、既存のPowerPC命令に対し、PowerPCソフトウェアが、メモリ中のどこかに位置するＸ86ソフトウェアに直接分岐することを許す拡張を可能にする。具体的には、すべてのPowerPC命令が４バイト境界上に位置しているため、PowerPC分岐命令中に記憶されたアドレスへと分岐する命令は、通常、レジスタの低位２ビットを無視する。しかし、Ｘ86命令はバイト整列されているため、PowerPC分岐は、１、２または３のバイト境界上にある命令に達することができない。Ｑビットは、前述の分岐アドレス・レジスタの低位２ビットを可能にし、それにより、PowerPCコードがバイト整列されたアドレスに分岐することを許すことにより、この問題を解消する。

【００８８】Ｑビットの制御の下でデコーダ803および804によって生成される共通の命令は、実行リソース、例えば整数装置（ＩＵ）106および浮動小数点装置（ＦＰＵ）107（図１）の動作に対して直接的な影響を及ぼす。Ｘ86デコーダ803は、ＧＰＲ1062（図１）およびＦＰＲ1072（図１）中のレジスタのサブセットのみを使用する共通の命令を生成することができ、一方、PowerPCデコーダ804は、すべてのレジスタを使用する共通の命令を生成することができる。さらに、Ｑビットは二つのアーキテクチャのコンテキストを制御するが、一方のアーキテクチャによって使用される多数のレジスタが他方のアーキテクチャによってアクセスされることができる。具体的には、PowerPCソフトウェアは、すべてのＸ86レジスタに読み書きすることができ、Ｘ86ソフトウェアは、PowerPCのＭＳＲ、ＢＡＴおよびデクレメンタ（d

ecrementer）・レジスタに読み書きすることができる。

【００８９】Ｘ86命令デコーダ803およびPowerPC命令デコーダ804から、デコードされた共通の形態の命令が、図８を参照しながら説明したように、Ｑビットによって制御される命令選択論理805を介して送出し論理806に移される。さらには、命令長の情報が次の命令取出しアドレス（ＮＩＦＡ）計算ブロック807に転送される。このＮＩＦＡ計算ブロック807を使用して、命令キャッシュ104（図１）から取り出さなければならない次の命令のアドレスを計算する。これは、その入力として、現在の命令のアドレスを命令取出しバスから取り込み、命令長を二つのアーキテクチャ・デコーダ803および804から取り込み、二つの命令長の間で選択を行うアーキテクチャ・モード制御信号を取り込む。分岐処理装置（ＢＰＵ）808（図１のＢＰＵ1054に相当）もまた、ＮＩＦＡ計算ブロック807への入力を有している。この入力は、分岐命令が次の命令取出しアドレスを分岐の目標に強いることを許す。

【００９０】図６に示す論理によって生成されるモード切換えは、デコーダ803および804に入力される。この信号が、「モード切換え」命令と呼ばれる疑似命令として作用し、命令の流れの中で位置設定記号として働く。現在のプログラムのコードが完了すると、他方のアーキテクチャ・コンテキストへのモード切換えが生じる。分岐予測装置（ＢＰＵ）803が、コード中のこの点を検出し、モード切換えを可能にして最終的に生じさせる遅延モード切換え（モード切換え′）信号を出力する。この信号はまた、ＭＵＸ67（図６）にもフィードバックされて、レジスタ66中のＱビットの状態を切り換え、モード切換えを反映させる。

【００９１】Ｑビットはまた、メモリ管理装置（ＭＭＵ）103（図１）にも入力されて、どちらのアドレス変換機構がマイクロプロセッサによって用いられるかを制御する。オンチップＭＭＵ103は、PowerPCまたはＸ86のいずれかのページ・テーブル項目を共通のフォーマットで記憶することができる一つのページ・テーブル・キャッシュ、すなわち変換ルックアサイド・バッファ（ＴＬＢ）を含む。Ｑビットの設定が、ＴＬＢ項目がいかにしてプロセッサのハードウェアによって初期化されるか、そして、それらがアドレス変換中にどのように解釈されるかを決定する。図１０は、マイクロプロセッサのＴＬＢ項目の共通のフォーマットならびにPowerPCおよびＸ86アーキテクチャのページ・テーブル項目がどのように中に記憶されているかを示す。

【００９２】本発明においては、アーキテクチャ修飾制御ビットＱが、両方のアーキテクチャによって使用されるアドレス変換機構を変更する。アーキテクチャ修飾制御ビットが１であるとき、オペレーティング・システムがPowerPCコードを含むメモリ・ページとＸ86コードを含むメモリ・ページとを区別することを可能にする、PowerPCページング機構に対する拡張が可能になる。拡張のルートは、各PowerPCページ・テーブル項目中の、各ページをPowerPCページ（ビットが０にセット）またはＸ86ページ（ビットが１にセット）のいずれかとして定義するページ・モード制御ビットＰからなる。このビットを使用して、上記で説明し、図７に表したようにして、アーキテクチャ・コンテキスト制御信号の値を駆動する。

【００９３】命令がメモリから取り出されるとき、まず、そのアドレスが、図１に示すＭＭＵ103によって変換される。ＭＭＵ103は、ページ・モード制御ビットＰの値を含む変換されたアドレスをもつ状態ビットを戻す。ページ・モード制御ビットの値がアーキテクチャ修飾制御ビットの現在値に等しいならば、プロセッサは以前と同様に同じアーキテクチャで作動を続ける。しかし、ページ・モード制御ビットの値がアーキテクチャ修飾制御ビットと異なるならば、以前に取り出されたすべての命令が実行され、例外なしに完了されるまで、命令取出しおよびデコードが停止される。完了したならば、アーキテクチャ・モード制御信号の値が、ページ・モード制御ビットの値に一致するように変更され、アーキテクチャ・モード制御信号によって確立された新たなコンテキストの下で命令取出しおよびデコードが再開される。

【００９４】PowerPC 変換機構に対する機能強化およびＸ86機構に対して加えられる限定が、特定的に、Ｘ86アドレス変換をPowerPC アドレス変換に対してマッピングすることを許して、両方のアーキテクチャ用に書かれたソフトウェアを多重タスク処理システム中で作動させるためのより動的な環境を提供する。アーキテクチャ修飾制御ビットＱが１にセットされると、PowerPCアドレス変換機構が「管理状態」になる。これが意味することは、Ｘ86セグメント変換が起こるが、Ｘ86ページ変換を実行する代わりに、PowerPCセグメントおよびページ変換が続いて起こってＰＡを形成するということである。これが、一つのオペレーティング・システムがいずれのアーキテクチャ用に書かれたソフトウェアのアドレス変換をも管理することを許す。唯一の制限は、Ｘ86ソフトウェアがそれ自体のＸ86ページング関連の作業を実行することができないことである。

【００９５】64ビットPowerPCの具現化は64ビット・アドレスを変換し、一方、今日定義されるＸ86アーキテクチャは32ビット・アドレスを生成する。本発明は、32ビット・レジスタ値を最終的な64ビット・アドレスの高位32ビットとして連結することにより、32ビットＸ86アドレスから64ビット・アドレスを生成するための手段を提供する。これが、32ビットＸ86アドレス空間を64ビット空間のどこかに位置づけることを許し、また、レジスタの値を動的に変化させることにより、オペレーティング・システムが多数のＸ86アドレス空間を管理することを

許す。32ビットの具現化においては、このレジスタ値は無視される（すなわち、実質的に０に強いられる）。

【００９６】図１１は、本発明による方法を使用して二つのアドレス変換機構をどのように併合するか、また、その処理のどこで32ビット・レジスタ値がX86アドレスと連結されて64ビット・アドレスを形成するのかを高レベルで示す。レジスタ1001が、X86論理アドレスを構成する32ビット・オフセットおよび16ビット・セレクタを保持する。このセレクタを使用して記述子テーブル1002をアドレス指定する。テーブル1002からのセグメント記述子が加算器1003中でオフセットと合わされて32ビットX86線形アドレスをレジスタ1004中に生成する。レジスタ1005中の32ビットX86基底アドレスが64ビットPowerPC実効アドレス・レジスタ1006の基底32ビットの中に読み込まれる。レジスタ1004中の32ビット線形アドレスがレジスタ1006中の32ビット基底アドレスと連結されて64ビットPowerPC実効アドレスを形成する。レジスタ1006からの有効セグメントＩＤを使用してセグメント・テーブル1007をアドレス指定して、図２を参照しながら説明したように、80ビットPowerPC仮想アドレスをレジスタ1008中に生成する。レジスタ1008の仮想セグメントＩＤを使用してページ・テーブル1009をアドレス指定して、図４を参照しながら説明したように、52ビットPowerPC実アドレスを生成する。

【００９７】図１のメモリ管理装置103はまた、アーキテクチャ・コンテキスト制御信号の下、PowerPCおよびX86アーキテクチャで、ページングされたメモリ保護検査を実行する責任を負う。図１２はPowerPCページ保護機構を示す。保護情報が三つのソースから収集される。すなわち、セグメント・テーブル項目1101（64ビットPowerPC）またはセグメント・レジスタ1101（32ビットPowerPC）中に見いだされるＫｓおよびＫｐセグメント保護ビット、PowerPC機械状態レジスタ（ＭＳＲ）1102中に見いだされるスーパバイザ／ユーザ・モード・ビット（ＰＲ）、そして、下寄りページ・テーブル項目1104中に見いだされるＰＰページ保護ビットである。保護キー1105は、ＫｐセグメントビットをＭＳＲ．ＰＲスーパバイザ／ユーザ・モード・ビットとでＡＮＤ演算し、その結果を、ＫｓセグメントビットとＭＳＲ．ＰＲスーパバイザ／ユーザ・モード・ビットの否定との論理積とでＯＲ演算することによって形成される。キー1105を使用して、メモリ管理装置103（図１）がページ保護ＰＰビットの値を検査して、テーブル1106に示すような許容されるアクセスのタイプを決定する。

【００９８】図１３は、X86ページ保護機構が作動する方法を示す。保護情報が四つのソースから収集される。すなわち、現在のスタック・セグメント記述子1201の記述子特権レベル（しばしば現在の特権レベルまたはＣＰＬと呼ぶ）、レジスタＣＲ０（1202）からの書込み保護ビットＷＰ、ページ・テーブル項目1203からのユーザ／スーパバイザ・ビットＵ／Ｓ、そして、同じくページ・テーブル項目1203からの読み／書きビットＲ／Ｗである。この情報を使用して、メモリ管理装置がテーブル1204に示すような検査を実行して、許容されるアクセスのタイプを決定する。

【００９９】本発明においては、アーキテクチャ修飾機構ビットは、プロセッサがX86モードで作動しているときに使用される保護機構のみを変更する。具体的には、X86ページ変換がPowerPCページ変換によって取って代わられているため、X86ページ保護機構はPowerPCページ保護機構によって完全に取って代わられる。X86モードでは、修飾機構ビットがPowerPC ＭＳＲ　ＰＲビットをしてX86ＣＰＬの特権を反映させる。０、１または２のX86ＣＰＬがＭＳＲ．ＰＲを０（スーパバイザ）の値に強要し、３のX86ＣＰＬがＭＳＲ　ＰＲビットを１（ユーザ）の値に強要する。すると、上記のように保護キー1105（図１１）が形成される。ＭＳＲＰＲビットをしてX86モードでＣＰＬを追跡させることにより、X86スーパバイザ・コードをX86およびPowerPCユーザ・コードから保護することができる。オペレーティング・システムのソフトウェアは通常、スーパバイザ・レベルで作動するため、これが、好都合なところで、オペレーティング・システムの一部をX86コードで具現化することを許す。

【０１００】図１４は、PowerPCアーキテクチャのうち、割込みおよび例外の動作を制御する部分のブロック図である。機械状態レジスタ1301を使用して、ＥＥビット1301によって外部割込みを可能にしたり不能にしたりし、また、ＩＰビット1311によって物理メモリ中の割込みベクトル・テーブル1307の場所を決定する。レジスタＳＲＲ０（1302）を使用して、結果的に例外となったか、割込みの場合に実行されていたであろう命令の実効アドレスを記録する。レジスタＳＲＲ１（1302）は、例外の特定の原因に関する状態情報ならびに割込みまたは例外の前のＭＳＲ1301中の特定のビットを含む。レジスタＤＡＲ1304は、データ関連の例外中の例外を引き起こしたデータ・オペランドの実効アドレスを保持する。レジスタＤＳＩＳＲ905は、データ関連の例外の特定の原因に関する状態情報を含む。

【０１０１】割込みまたは例外がPowerPCアーキテクチャによって受け入れられるとき、割込み手順の場所は、割込みのタイプおよびＭＳＲ割込みプレフィックス（ＩＰ）ビット1311の値に依存する。ＭＳＲ　ＩＰビット1311は、割込みベクトルテーブル基底アドレス1306が0x00000000の値を有するのか、0xFFF00000の値を有するのかを指定する。このアドレスが、タイプ割込みによって指定されるベクトル・テーブル1307へのオフセットに加えられて、割込み手順1309の物理アドレスを形成する。例えば、ＩＶＴ基底アドレス1306が0xFFF00000であり、データ記憶割込みが受け入れられるならば、割込み

手順1309の物理アドレスは0ｘFFF00300になる。

【０１０２】図１５は、割込み手順場所をＸ86アーキテクチャによって指定する二つの方法を示す。いずれの方法においても、割込み番号1401は、命令、外部的割込みまたは内部的例外のいずれかによって指定される。Ｘ86実モード（保護モードは不能になる）では、割込みベクトル・テーブル1402は普通、0ｘ00000000の基底アドレスを有している。割込み手順は、割込み番号1401に４を掛け、それにより、割込みプロセッサに対する「遠い」ポインタを提供することによって位置づけられる。この「遠い」ポインタは、オフセットおよびセグメント値からなる。Ｘ86保護モード（アドレス変換は可能になる）では、割込み番号1401に８を掛けて、割込み記述子テーブル1403へのオフセットを出す。基準化された割込み番号1401によって参照される割込みゲートは、宛先コード・セグメント1405へのオフセットおよびセグメント・テーブル1406へのオフセットを含む。オフセットによってセグメント・テーブル1406中に指されるセグメント記述子1407は、宛先コード・セグメント1405の基底アドレスを含む。この基底アドレスが宛先コード・セグメント1405へのオフセットに加算されて、割込み手順1408の実効アドレスを形成する。

【０１０３】Ｘ86アーキテクチャにおいて割込みが受け入れられると、図１６に示すように、特定の情報が割込み手順スタック上に記録される。これは、割込みを受けた手順のスタックポインタ（旧ＳＳ1501および旧ＥＳＰ1502）と、割り込まれた命令へのポインタ（旧ＣＳ1504および旧ＥＩＰ1505）と、エラー・コード1506と、割り込まれた命令の状態を含むＥＦＬＡＧＳ1503のコピーとからなる。加えて、ページ障害割込みが、ＣＲ２（1507）においてページ障害を生じさせた命令またはデータ・バイトのアドレスを記憶する。

【０１０４】PowerPCおよびＸ86の両アーキテクチャの割込みおよび例外は、図１の分岐処理装置（ＢＰＵ）1054によって取り扱われる。アーキテクチャ・モード制御装置108からのアーキテクチャ・コンテキスト機構ビットが、二つの割込みおよび例外機構、すなわちPowerPCまたはＸ86のどちらがマイクロプロセッサによって使用されるかを決定する。

【０１０５】アーキテクチャ修飾機構ビットが可能になると、プロセッサによって遭遇される割込みおよび例外は、割込みまたは例外のタイプならびにアーキテクチャ・コンテキスト機構ビットの状態に依存して、PowerPCまたはＸ86アーキテクチャ型機構のいずれかによって取り扱われる。具体的には、アーキテクチャ・コンテキスト機構ビットの状態にかかわらず、すべての非同期的割込みがPowerPC割込み機構に向けられる。これは、オペレーティング・システムが、プロセッサが実行していることに関連がないかもしれない事象が起こったとき、そのような事象に対して一貫した制御ポイントを有することを許すために行われる。

【０１０６】同期的例外は、一般に、アーキテクチャ・コンテキスト機構ビットによって定義されるアーキテクチャ型機構に向けられる。しかし、これは、（ａ）ページ変換から生じる形態の例外および（ｂ）ＭＭＵ内で生じるPowerPC保護検査から生じる形態の例外の場合には、真ではない。いずれの場合においても、アーキテクチャ・コンテキスト機構ビットの値にかかわらず、例外はPowerPC機構に向けられる。理由は、Ｘ86ページングが動かされないため、すべてのPowerPC ＭＭＵ関連の例外をPowerPC機構に転送しなければならないからである。

【０１０７】一つの好ましい実施態様に関して本発明を説明したが、当業者であれば、請求の範囲の真髄および範囲において本発明に変更を加えうることを認識するであろう。

【０１０８】まとめとして、本発明の構成に関して以下の事項を開示する。

（１）別個の命令セットおよびメモリ管理方式を有する第一および第二のアーキテクチャをサポートし、一つの多重タスク処理オペレーティング・システムの下で作動するプロセッサにおいて、前記第一のアーキテクチャの第一の命令セットの命令をデコードし、前記第二のアーキテクチャの第二の命令セットの命令を直接デコードし、前記第一の命令セットのデコードされた命令を前記第二の命令セットの一つ以上の命令に対してマッピングするための命令セット管理手段と、前記第一および第二のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行するためのメモリ管理手段と、メモリから読み出されるプログラムのアーキテクチャ・コンテキストを、前記第一のアーキテクチャのコードまたは前記第二のアーキテクチャのコードのいずれかとして検出し、前記命令セット管理手段および前記メモリ管理手段を制御して、前記第一のアーキテクチャのアドレス変換と第二のアーキテクチャのアドレス変換との間で動的に切り換え、前記第二のアーキテクチャの、一つ以上のマッピングされてデコードされた命令または直接デコードされた命令を実行するための制御手段と、を含むことを特徴とするプロセッサ。

（２）前記メモリ管理手段が、実効アドレスを仮想アドレスに変換する際に第一または第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するための、前記制御手段によって制御されるモード制御機構を含む上記（１）記載のプロセッサ。

（３）前記モード制御機構が前記プロセッサの命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が、前記命令セット管理手段による適切なデコードのために取り出され、配列されるようにする上記（２）記載のプロセッサ。

（４）前記メモリ管理装置が、前記第二のアーキテクチ

ャのページ・テーブル項目からページ・モード制御ビットを読み出し、前記ページ・モード制御ビットが前記制御手段に供給されて、前記メモリ管理手段によるアドレス変換を制御する上記（３）記載のプロセッサ。

（５）前記制御手段が、プロセッサがどちらのアーキテクチャ・コンテキストの下で作動するかを制御するためのアーキテクチャ・コンテキスト制御機構であって、実効アドレスを仮想アドレスに変換する際に前記プロセッサの前記メモリ管理装置によって使用されるアーキテクチャ変換方法を制御し、さらに、前記プロセッサの命令取出しおよびデコード論理を制御し、さらに、プロセスおよび構築されたリソースの実行コンテキストを制御し、さらに、前記プロセッサの割込みおよび例外機構を制御するためのアーキテクチャ・コンテキスト制御機構と、一つのアドレス変換機構が、一方のアーキテクチャのアドレスをもう一方のアーキテクチャのアドレスの変換に対してマッピングすることを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびの保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするための修飾モード制御機構と、を含む上記（１）記載のプロセッサ。

（６）前記命令セット管理手段が、前記第一の命令セットの命令をデコードするための第一のデコード手段と、前記第二の命令セットの命令をデコードするための第二のデコード手段と、前記第一のデコード手段からのデコードされた命令を前記第二の命令セットの一つ以上のデコードされた命令に対してマッピングするためのマッピング手段と、前記マッピング手段からのデコードされた命令または前記第二のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される選択手段と、を含む上記（１）記載のプロセッサ。

（７）前記第一および第二のデコード手段が前記第一および第二のアーキテクチャの簡単な命令をデコードし、簡単な命令が、基本的な演算クラスの中にマッピングし、一つの実行装置によって取り扱われることができる命令であり、前記命令管理手段がさらに、前記第一の命令セットの複雑な命令をデコードするための第三のデコード手段と、前記第二の命令セットの複雑な命令をデコードするための第四のデコード手段と、前記第三または第四のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される第二の選択手段と、前記第三または第四のデコード手段からのデコードされた命令を前記第二の命令セットの多数の簡単な命令に対してマッピングするための、前記第二の選択手段の出力を受ける第二のマッピング手段と、複雑な命令がデコードされる場合には前記第二のマッピング手段の出力を選択し、簡単な命令がデコードされる場合には前記第一の選択手段の出力を選択するための、前記第三または第四のデコード手段の一方からの有効な信号に応答する第三の選択手段と、をさらに含む上記（６）記載のプロセッサ。

（８）前記制御装置が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前記プロセッサが複数の状態の一つに入るように制御するモード制御装置を含む上記（１）記載のプロセッサ。

（９）前記プロセッサが、主メモリから読み出される命令のページ・モードを検出し、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを前記第二のビットによって前記モード制御装置に知らせるメモリ管理装置を含む上記（８）記載のプロセッサ。

（１０）前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする上記（８）記載のプロセッサ。

（１１）一つの多重タスク処理オペレーティング・システムの下で作動するプロセッサにおいて具現化される、二つの別個の命令セット・アーキテクチャをサポートする方法であって、ソフトウェアによって供給されるモード制御ビットに応答して実効アドレスを仮想アドレスに変換する際に第一および第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するステップと、プロセッサの命令取出しおよびデコード機構を制御して、二つの異なるアーキテクチャの命令がソフトウェアによって供給されるモード制御ビットに応答して適切にデコードされるようにするステップと、一方のアーキテクチャの変換をもう一方のアーキテクチャのそれに対してマッピングすることにより、前記二つの異なるアーキテクチャのアドレスを変換するステップと、多重タスク処理環境において、一方のアーキテクチャ用に書かれたアプリケーション・ソフトウェアからもう一方のアーキテクチャ用に書かれたアプリケーション・ソフトウェアに切り換えるステップと、を含むことを特徴とする方法。

（１２）マイクロプロセッサの割込みおよび例外機構を制御するステップと、一つのアドレス変換機構が、第一のアーキテクチャのアドレスを第二のアーキテクチャの変換に対してマッピングすることによって第一のアーキ

テクチャのアドレスを変換することを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびに保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするステップと、命令が実行すべきところのアーキテクチャ・コンテキストを決定するステップと、をさらに含む上記（１１）記載のプロッセッサにおいて具現化される方法。

（１３）二つの別個の命令セット・アーキテクチャをサポートし、一つの多重タスク処理オペレーティング・システムの下で作動するマイクロプロセッサと、前記二つのアーキテクチャ用のアプリケーション・ソフトウェアを記憶する外部メモリ・デバイスと、前記マイクロプロセッサを前記外部メモリ・デバイスに接続するシステム・バスとを含み、前記マイクロプロセッサが、前記システム・バスに接続された内部バスを有し、前記マイクロプロセッサが、前記第一のアーキテクチャの第一の命令セットの命令をデコードし、前記第二のアーキテクチャの第二の命令セットの命令を直接デコードし、第一の命令セットのデコードされた命令を前記第二の命令セットの一つ以上の命令に対してマッピングするための命令セット管理手段と、前記第一および第二のアーキテクチャについて仮想アドレスから実アドレスへのアドレス変換を実行するためのメモリ管理手段と、メモリから読み出されるプログラムのアーキテクチャ・コンテキストを、前記第一のアーキテクチャのコードまたは前記第二のアーキテクチャのコードのいずれかとして検出し、前記命令セット管理手段および前記メモリ管理手段を制御して、前記第一のアーキテクチャのアドレス変換と前記第二のアーキテクチャのアドレス変換との間で動的に切り換え、前記第二のアーキテクチャの、一つ以上のマッピングされてデコードされた命令または直接デコードされた命令を実行するための制御手段と、を含むことを特徴とするコンピュータ・システム。

（１４）前記マイクロプロセッサの前記メモリ管理手段が、実効アドレスを仮想アドレスに変換する際に第一または第二のアーキテクチャ変換方法のどちらがメモリ管理装置によって使用されるかを制御するためのモード制御機構を含む上記（１３）記載のコンピュータ・システム。

（１５）前記モード制御機構が前記プロセッサの命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が、前記命令セット管理手段による適切なデコードのために取り出され、配列されるようにする上記（１４）記載のコンピュータ・システム。

（１６）前記メモリ管理装置が、前記第二のアーキテクチャのページ・テーブル項目からページ・モード制御ビットを読み出し、前記ページ・モード制御ビットが前記制御手段に供給されて、前記メモリ管理手段によるアドレス変換を制御する上記（１５）記載のコンピュータ・システム。

（１７）前記制御手段が、プロセッサがどちらのアーキテクチャ・コンテキストの下で作動するかを制御するためのアーキテクチャ・コンテキスト制御機構であって、実効アドレスを仮想アドレスに変換する際に前記マイクロプロセッサのメモリ管理装置によって使用されるアーキテクチャ変換方法を制御し、さらに、前記マイクロプロセッサの命令取出しおよびデコード論理を制御し、さらに、プロセスおよび構築されたリソースの実行コンテキストを制御し、さらに、前記マイクロプロセッサの割込みおよび例外機構を制御するためのアーキテクチャ・コンテキスト制御機構と、一つのアドレス変換機構が、一方のアーキテクチャのアドレスをもう一方のアーキテクチャのアドレス変換に対してマッピングすることを許容し、統合された割込みおよび例外機構が、割込みまたは例外が生じた際に実効中のアーキテクチャ・コンテキストにかかわりなく、非同期的な割込みおよびページ変換ならびに保護関連の例外を取り扱うことを許容する拡張および限定を前記二つのアーキテクチャに対して可能にするための修飾モード制御機構と、を含む上記（１３）記載のコンピュータ・システム。

（１８）前記マイクロプロセッサの前記命令セット管理手段が、前記第一の命令セットの命令をデコードするための第一のデコード手段と、前記第二の命令セットの命令をデコードするための第二のデコード手段と、前記第一のデコード手段からのデコードされた命令を前記第二の命令セットの一つ以上のデコードされた命令に対してマッピングするためのマッピング手段と、前記マッピング手段からのデコードされた命令または前記第二のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される選択手段と、を含む上記（１３）記載のコンピュータ・システム。

（１９）前記第一および第二のデコード手段が前記第一および第二のアーキテクチャの簡単な命令をデコードし、簡単な命令が、基本的な演算クラスの中にマッピングし、一つの実行装置によって取り扱われることができる命令であり、前記命令管理手段がさらに、前記第一の命令セットの複雑な命令をデコードするための第三のデコード手段と、前記第二の命令セットの複雑な命令をデコードするための第四のデコード手段と、前記第三または第四のデコード手段からのデコードされた命令を選択するための、前記制御手段によって制御される第二の選択手段と、前記第三または第四のデコード手段からのデコードされた命令を前記第二の命令セットの多数の簡単な命令に対してマッピングするための、前記第二の選択手段の出力を受ける第二のマッピング手段と、複雑な命令がデコードされる場合には前記第二のマッピング手段の出力を選択し、簡単な命令がデコードされる場合には

前記第一の選択手段の出力を選択するための、前記第三または第四のデコード手段の一方からの有効な信号に応答する第三の選択手段と、を含む上記（１８）記載のコンピュータ・システム。

（２０）前記マイクロプロセッサの前記制御手段が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前記プロセッサが複数の状態の一つに入るように制御するモード制御装置を含む上記（１３）記載のコンピュータ・システム。

（２１）前記プロセッサが、主メモリから読み出される命令のページ・モードを検出し、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを前記第二のビットによって前記モード制御装置に知らせるメモリ管理装置を含む上記２０記載のコンピュータ・システム。

（２２）前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする上記２０記載のコンピュータ・システム。

（２３）前記マイクロプロセッサが、サポートされる二つのアーキテクチャに共通のフォーマットを使用して具現化される一つのメモリ管理装置を有している上記１３記載のコンピュータ・システム。

（２４）前記マイクロプロセッサが、前記サポートされる二つのアーキテクチャによって共用される一つの命令取出し機構と、別々の命令デコード機構とを有している上記１３記載のコンピュータ・システム。

（２５）前記マイクロプロセッサのすべての実行リソースが前記サポートされる二つのアーキテクチャに共通である上記１３記載のコンピュータ・システム。

（２６）別個の命令セットおよびメモリ管理方式を有する第一および第二のアーキテクチャをサポートするプロセッサにおいて、前記第一および第二のアーキテクチャ用に書かれたアプリケーション・プログラムを記憶する外部メモリに接続するためのシステム・インタフェースと、前記第二のアーキテクチャのページ・テーブル項目からページ・モード制御ビットを読み出すための、前記システム・インタフェースに接続されたメモリ管理装置と、第一の命令セットの命令をデコードし、デコードされた命令を第二の命令セットの一つ以上のデコードされた命令に対してマッピングするための第一のデコード手段と、第二の命令セットの命令をデコードするための第二のデコード手段と、前記第一または第二のデコード手段によってデコードされた命令を選択するための選択手段とを含む取出しおよびデコード機構を含む命令装置と、第一および第二のアーキテクチャ・モードのどちらを可能にするかに依存してアクセスすることができる複数のレジスタを有する、前記命令装置からのデコードされた命令を受信し、実行するために接続された実行装置と、前記命令装置および前記実行装置に接続されたモード制御装置とを含み、前記メモリ管理装置によって読み出される前記ページ・モード制御ビットが前記モード制御装置に供給され、前記第一または第二のアーキテクチャ・モードのいずれかを可能にしてアドレス変換を制御し、前記プロセッサの前記命令取出しおよびデコード機構を制御して、前記第一および第二のアーキテクチャの命令が適切なデコードに備えて取り出され、配列されるようにすることを特徴とするプロセッサ。

（２７）前記モード制御装置が、前記プロセッサ内のレジスタ中に記憶された、前記第二のアーキテクチャのアドレス変換を可能にするか不能にするかを決定する第一のビット、メモリ中の現在のページが前記第一のアーキテクチャのものか前記第二のアーキテクチャのものかを示す第二のビットおよびアーキテクチャ修飾ビットである第三のビットを含む複数のビットの状態に依存して、前記プロセッサが複数の状態の一つに入るように制御する上記２６記載のプロセッサ。

（２８）前記モード制御装置が、機能制御レジスタをさらに含み、前記プロセッサの起動またはリセット状態に応答して、前記プロセッサを前記第一および第二のアーキテクチャにそれぞれ対応する前記第一または第二のモードのいずれかで初期化し、その後、前記機能制御レジスタ中のビットの状態がアーキテクチャ・コンテキスト間での動的な切換えを可能にする上記２７記載のプロセッサ。

【図面の簡単な説明】

【図１】本発明を具現化することができるマイクロプロセッサのブロック図である。

【図２】PowerPCアーキテクチャにおける実効アドレスから仮想アドレスへの変換（セグメント化）を示すブロック図である。

【図３】Ｘ86アーキテクチャにおける実効アドレスから仮想アドレスへの変換（セグメント化）を示すブロック図である。

【図４】PowerPCアーキテクチャにおける仮想アドレスから物理アドレスへの変換（ページング）を示すブロック図である。

【図５】Ｘ86アーキテクチャにおける仮想アドレスから物理アドレスへの変換（ページング）を示すブロック図である。

【図６】図１の例示的なモード制御装置の関連の論理を

41

示す論理図である。

【図７】アーキテクチャ・モード制御装置がアーキテクチャ・コンテキスト制御を生成する方法を示す状態図である。

【図８】マイクロプロセッサの命令装置のデータの流れを示す高レベルブロック図である。

【図９】マイクロプロセッサの命令装置のブロック図である。

【図１０】変換ルックアサイド・バッファ（ＴＬＢ）フォーマットをPowerPCおよびＸ86のページのテーブル項目について対比させるブロック図である。

【図１１】本発明にしたがってＸ86アドレス変換がPowerPCアドレス変換にマッピングされることを示すブロック図である。

【図１２】PowerPCのページングされた保護検査規則を示すテーブルおよびブロック図のセットである。

【図１３】Ｘ86のページングされた保護検査規則を示すテーブルおよびブロック図のセットである。

42

【図１４】PowerPCの割込み状態および制御レジスタならびに割込みベクトル・テーブルを　示すブロック図である。

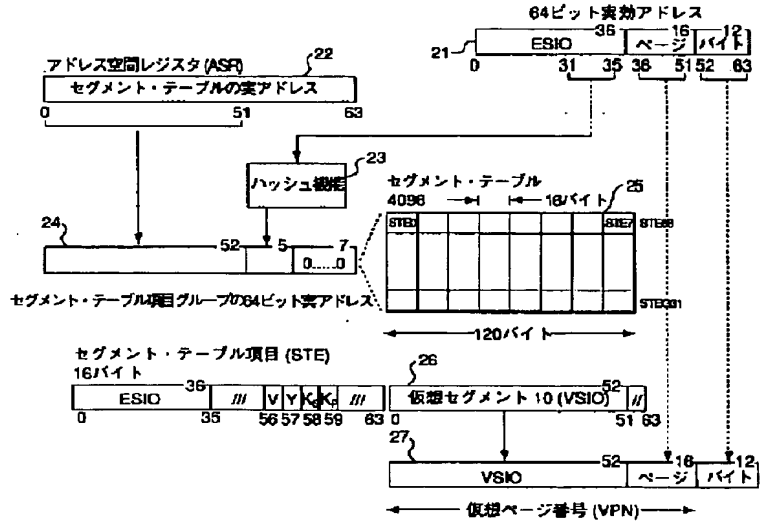
【図１５】Ｘ86の実モードおよび保護モードの割込みベクトル・テーブルを示すブロック図である。

【図１６】Ｘ86の割込み状態および制御レジスタならびに割込みスタックを示すブロック図である。

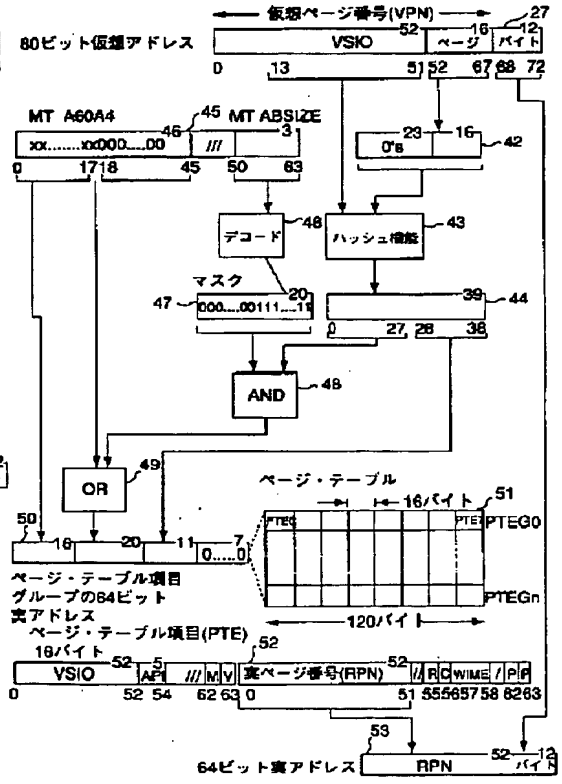
【符号の説明】

10	マイクロプロセッサ
12	システム・バス
101	システム・インタフェース
102	メモリ装置
103	メモリ管理装置
104	キャッシュ
105	命令装置
106	整数装置
107	浮動小数点装置
108	モード制御装置

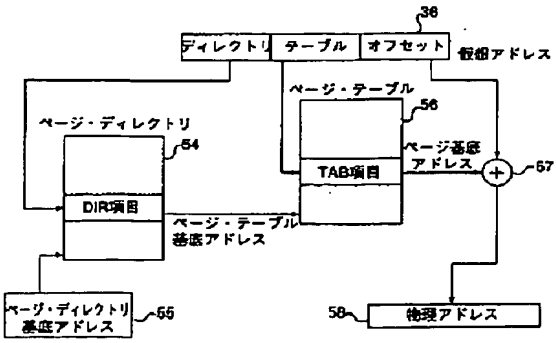
【図２】



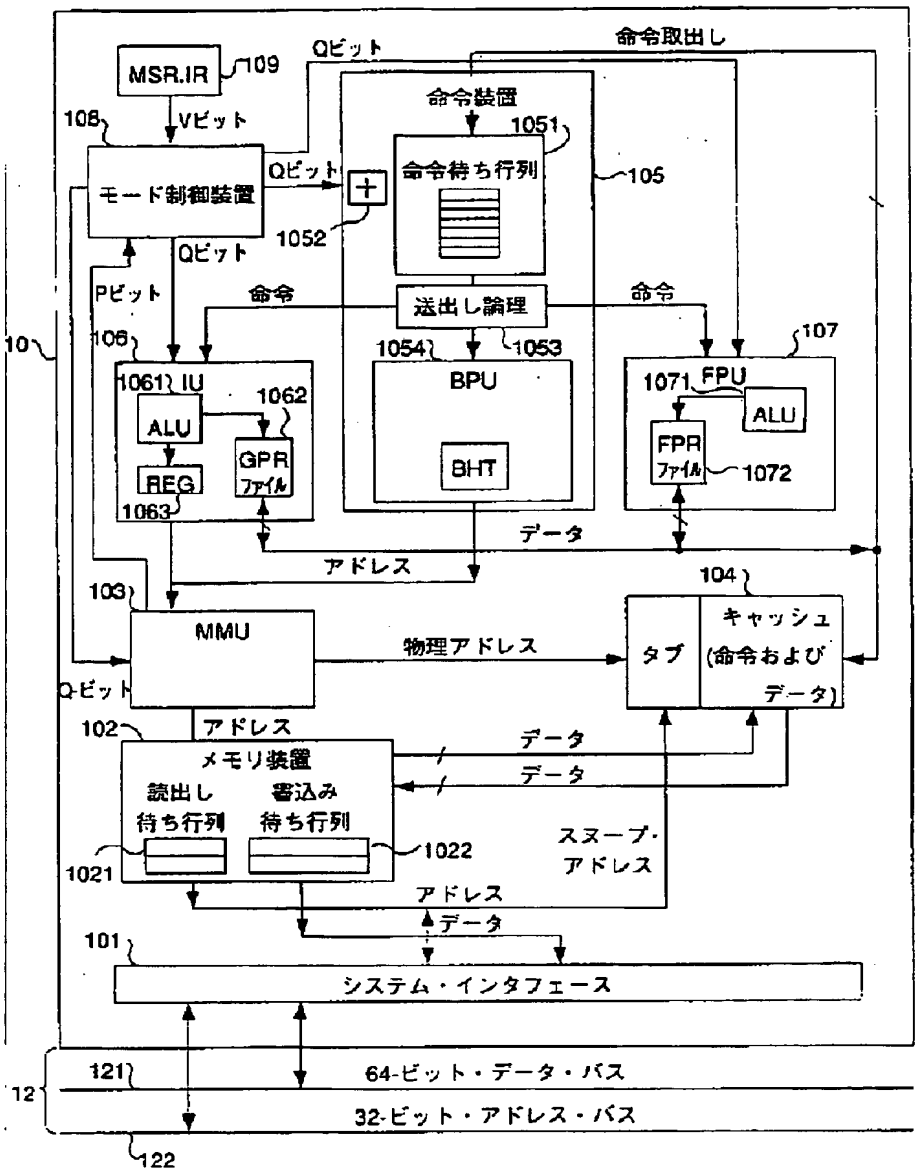
【図４】



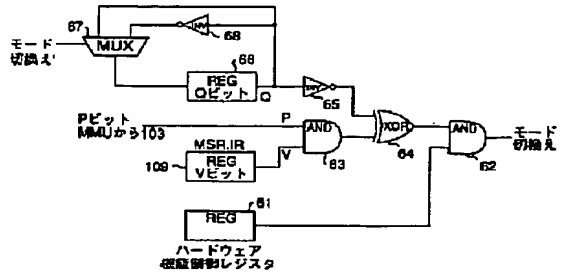
【図５】



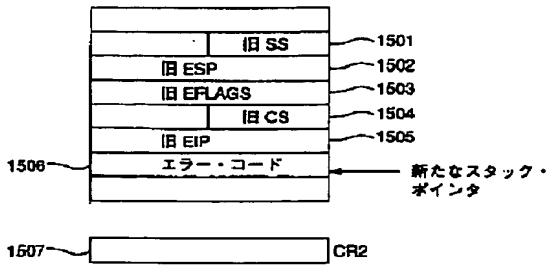
【図１】


MSR.IR
109
Qビット
命令取出し
Vビット
108
命令装置
1051
モード制御装置
Qビット
命令待ち行列
105
1052
Pビット
Qビット
命令
送出し論理
命令
10
106
107
1061 IU
1062
1054
1053
BPU
1071 FPU
ALU
ALU
FPR
ファイル
1072
REG
GPR
ファイル
BHT
1063
データ
アドレス
103
104
MMU
物理アドレス
タブ
キャッシュ
(命令および
データ)
Qビット
アドレス
102
データ
メモリ装置
データ
読出し
待ち行列
書込み
待ち行列
1022
スヌープ・
アドレス
1021
アドレス
データ
101
システム・インタフェース
121
64-ビット・データ・バス
12
32-ビット・アドレス・バス
122

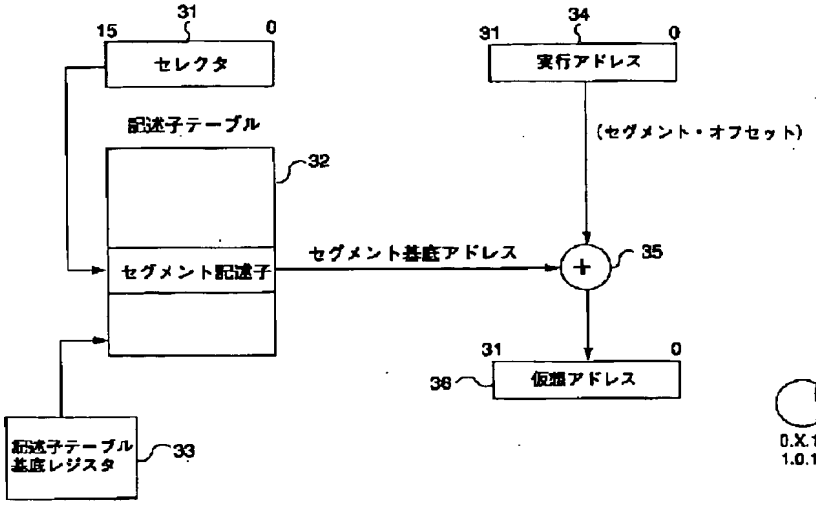
【図６】


67
MUX
68
モード
切換え
68
REG
Qビット
Q
65
Pビット
MMUから103
P
AND
XOR
AND
モード
切換え
MSR.IR
REG
Vビット
109
V
63
64
62
61
REG
ハードウェア

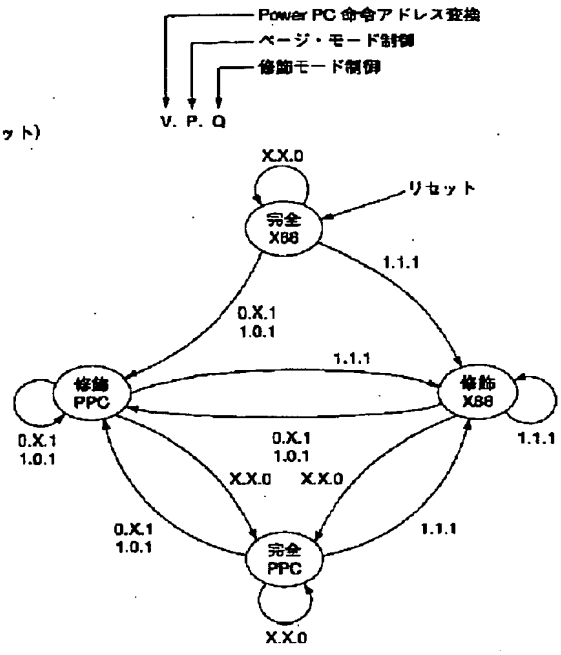
【図１６】


旧 SS
1501
旧 ESP
1502
旧 EFLAGS
1503
旧 CS
1504
旧 EIP
1505
1506
エラー・コード
新たなスタック・
ポインタ
1507
CR2

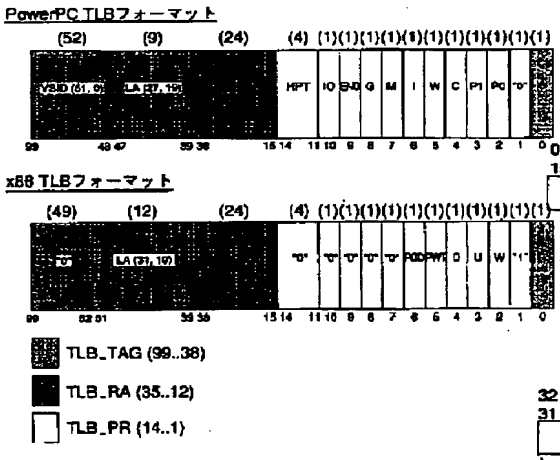
【図３】


31
15
0
セレクタ
記述子テーブル
32
セグメント記述子
記述子テーブル
基底レジスタ
33
34
31
0
実行アドレス
(セグメント・オフセット)
セグメント基底アドレス
35
31
0
36
仮想アドレス

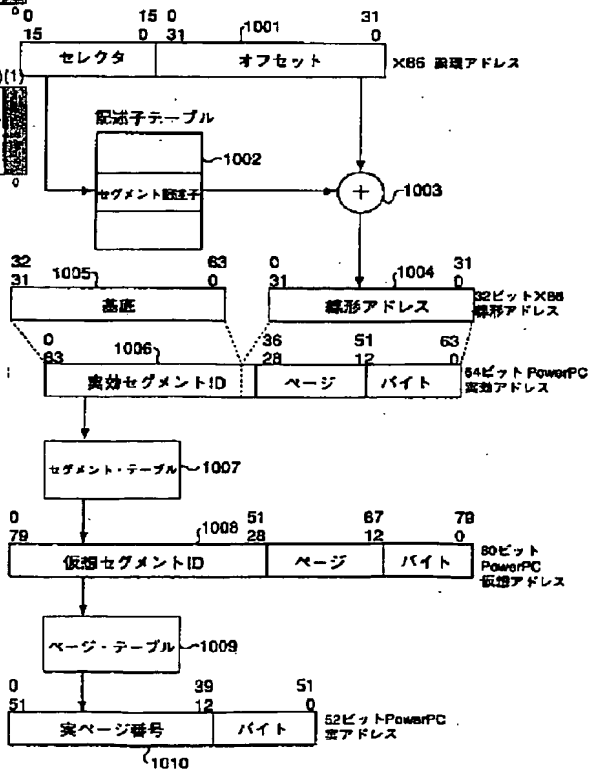
【図７】


Power PC 命令アドレス変換
ページ・モード制御
修飾モード制御
V. P. Q
X.X.0
リセット
完全
X86
1.1.1
0.X.1
1.0.1
修飾
PPC
修飾
X86
完全
PPC

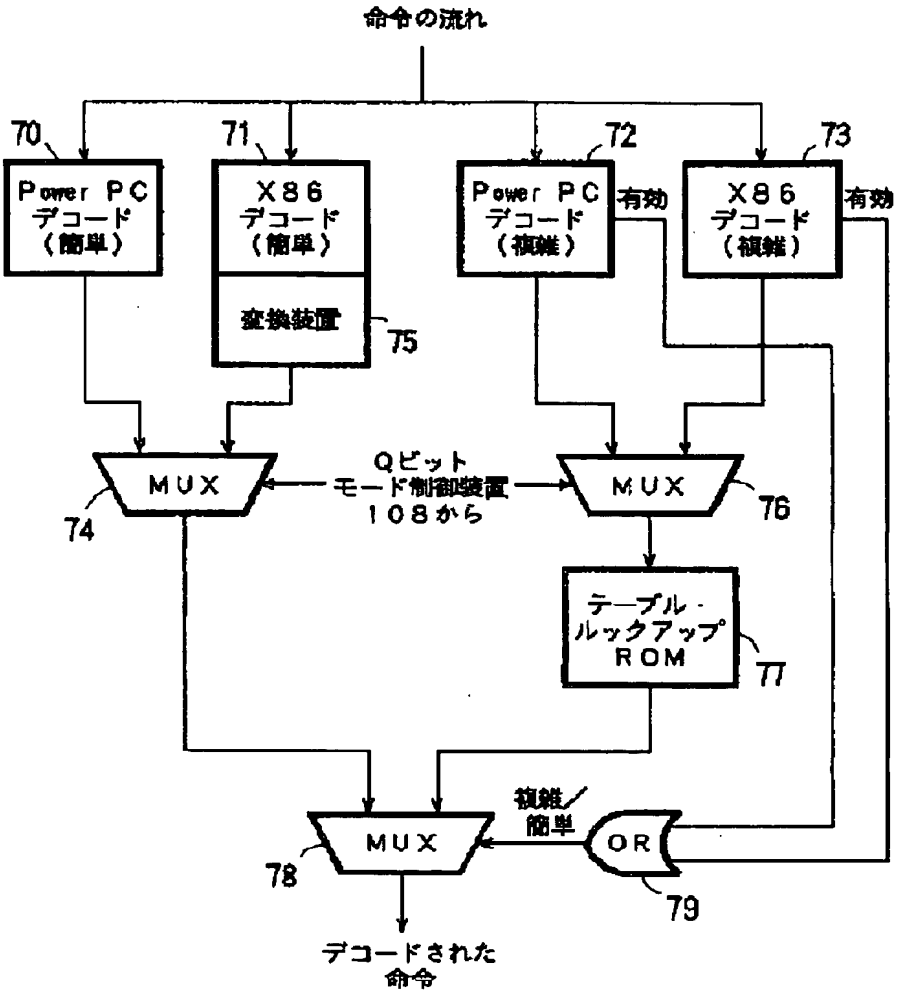
【図１０】


PowerPC TLBフォーマット
(52)
(9)
(24)
(4)
HPT
x86 TLBフォーマット
(49)
(12)
(24)
(4)
TLB_TAG (99..38)
TLB_RA (35..12)
TLB_PR (14..1)

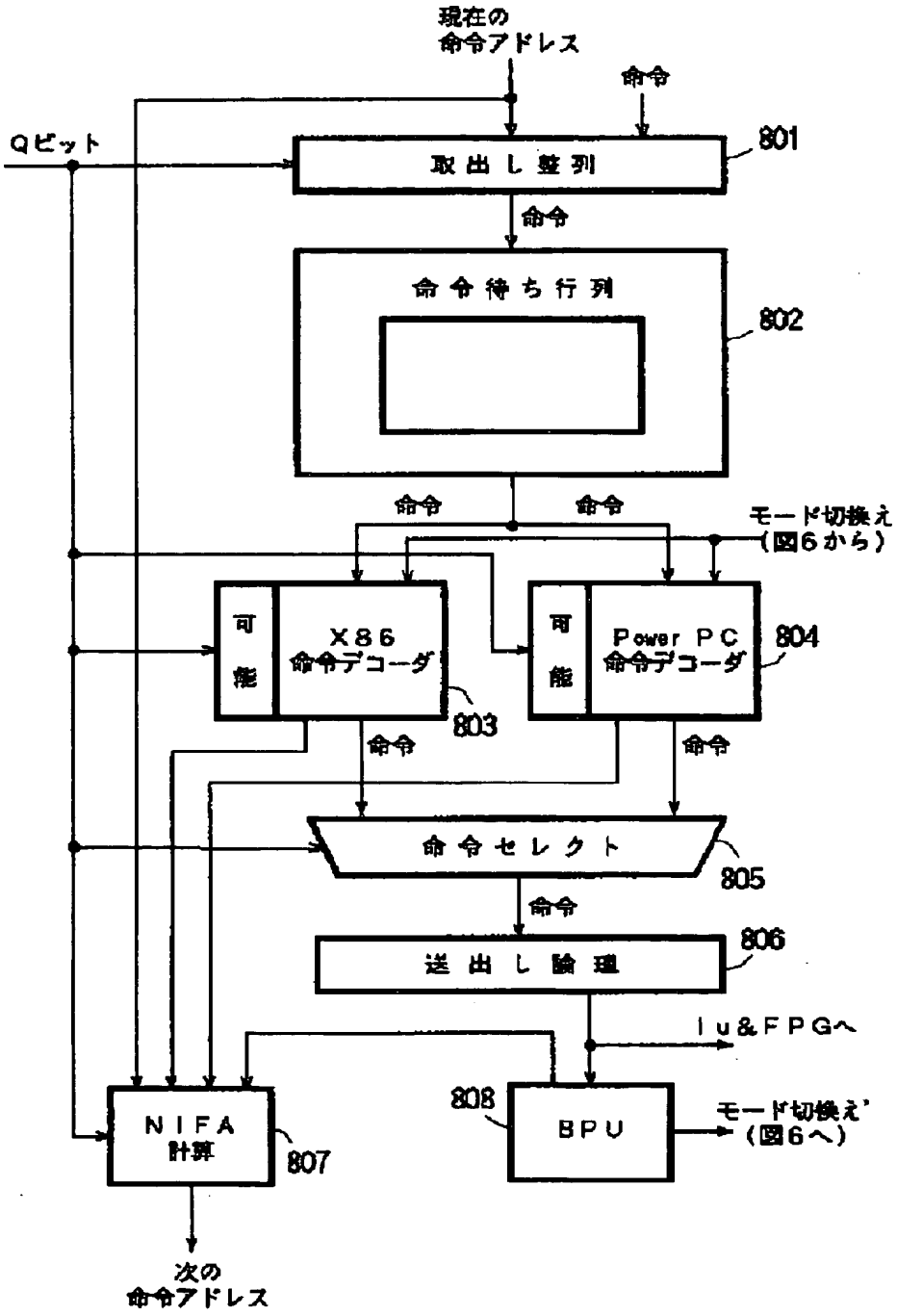
【図１１】


セレクタ
オフセット
X86 論理アドレス
記述子テーブル
1002
セグメント記述子
1003
基底
線形アドレス
32ビットX86
線形アドレス
実効セグメントID
ページ
バイト
64ビット PowerPC
実効アドレス
セグメント・テーブル
1007
仮想セグメントID
80ビット
PowerPC
仮想アドレス
ページ・テーブル
1009
実ページ番号
バイト
52ビットPowerPC
実アドレス
1010

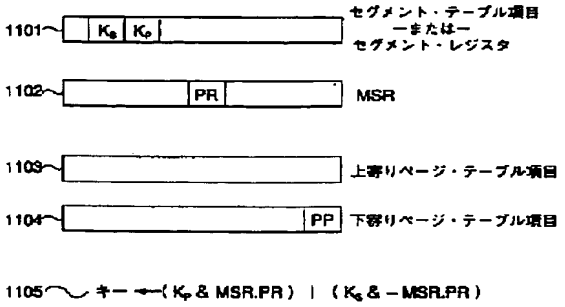
【図８】


命令の流れ
70
71
72
73
Power PC
デコード
（簡単）
X86
デコード
（簡単）
変換装置
75
Power PC
デコード
（複雑）
有効
X86
デコード
（複雑）
有効
MUX
74
Qビット
モード制御装置
108から
MUX
76
テーブル・
ルックアップ
ROM
77
複雑/
簡単
MUX
78
OR
79
デコードされた
命令

【図９】


現在の
命令アドレス
命令
Qビット
801
取出し整列
命令
命令待ち行列
802
命令
命令
モード切換え
(図6から)
可能
X86
命令デコーダ
803
可能
Power PC
命令デコーダ
804
命令
命令
命令セレクト
805
命令
806
送出し論理
Iu&FPGへ
808
BPU
モード切換え'
(図6へ)
NIFA
計算
807
次の
命令アドレス

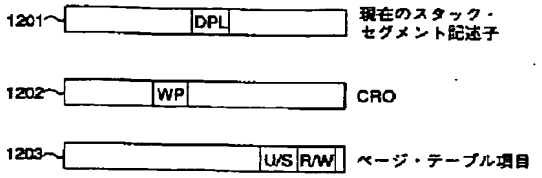
【図１２】



1106

キー	PP	アクセス許可
0	00	読み/書き
0	01	読み/書き
0	10	読み/書き
0	11	読出し専用
1	00	なし
1	01	読出し専用
1	10	読み/書き
1	11	読出し専用

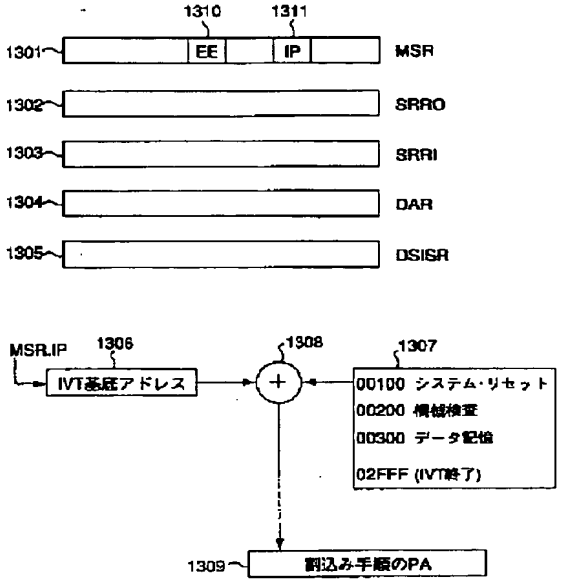
【図１３】



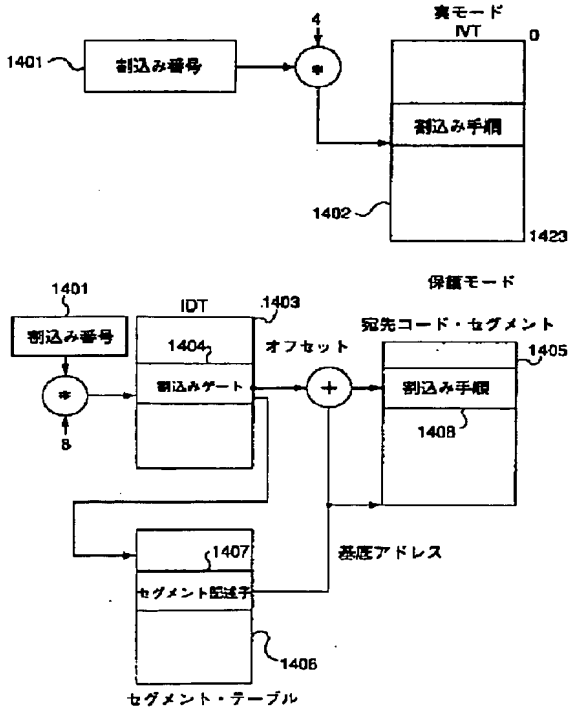
1204

WP	CPL	U/S	R/W	アクセス
0	ユーザ	0	0	読出し専用
0	ユーザ	0	1	R/W
0	ユーザ	1	0	なし
0	ユーザ	1	1	なし
0	スーパバイザ	0	0	R/W
0	スーパバイザ	0	1	R/W
0	スーパバイザ	1	0	読出し専用
0	スーパバイザ	1	1	R/W
1	ユーザ	0	0	読出し専用
1	ユーザ	0	1	R/W
1	ユーザ	1	0	なし
1	ユーザ	1	1	なし
1	スーパバイザ	0	0	読出し専用
1	スーパバイザ	0	1	R/W
1	スーパバイザ	1	0	読出し専用
1	スーパバイザ	1	1	R/W

【図１４】



【図１５】



フロントページの続き

(72)発明者　ステファン・ダブル・マヒン
アメリカ合衆国０５４８９、ヴァーモント州アンダーヒル　スティーブンスビル・ロード　アールアールワン　ボックス　１３９５
(72)発明者　ジョン・ジェイ・ヴァーグヴィスト
アメリカ合衆国０５４９５、ヴァーモント州ウィルストン　メットカーフ・ドライブ　１１４